ZX COMPUTING MONTHLY

AN ARGUS SPECIALIST PUBLICATION MAY 1987 £1.50

FOR ALL SINCLAIR USERS



THE PROFESSIONAL ADVENTURE WRITING SYSTEM

Extended Basic

128 Printer Toolkit

Supa-Plot: Printer Utility

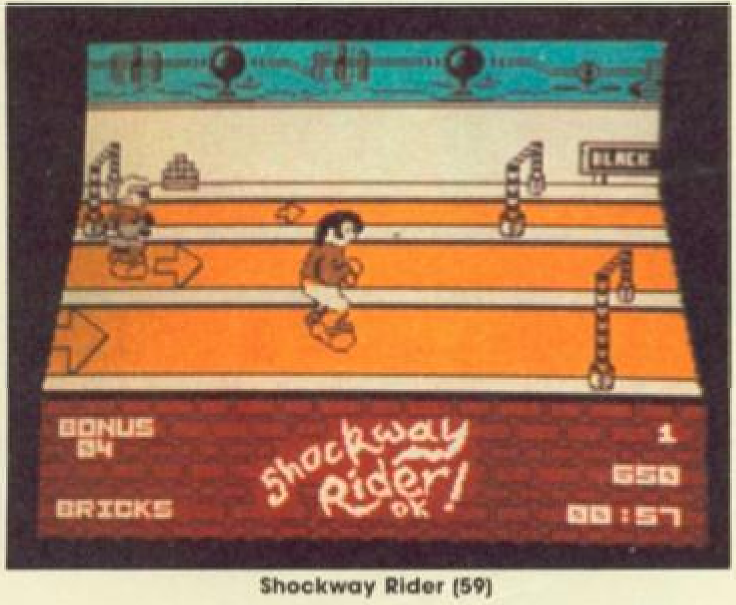
TRAPPED IN THE JUNGLES OF TERROR
YOU HEAR A RUSTLING SOUND BEHIND YOU.
WHAT NEXT???
READ THE P.A.W.S REVIEW - PAGE 13

MONSTER HITS! Short Circuit ▲ Legions of Death ▲ Tobruk

WIN! Indoor Sports ▲ SPECIAL OFFER: Lothlorien Wargames

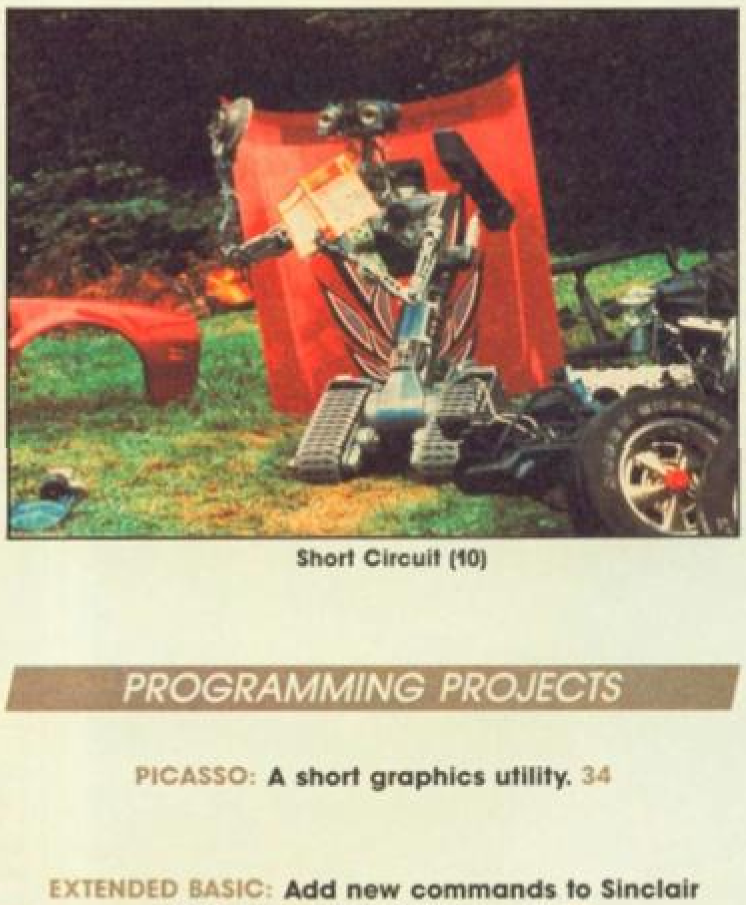
CONTENTS

FEATURES



Shockway Rider (59)

REGULARS



Short Circuit (10)

PROGRAMMING PROJECTS

SOFTWARE REVIEWS

NEWS

Cheetah on the move

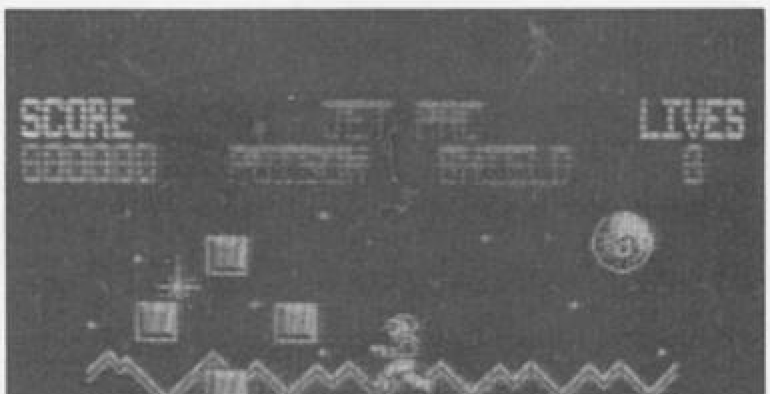
Leopards can't change their spots but Cheetah can change their address. By the time you read this, Cheetah Marketing should be firmly ensconsed in their new premises at Norbury House, Norbury Rd, Cardiff (Tel: 0222-555525).

The move is due to the fact that Cheetah are raking it in from the success of the SpecDrum and various other products and are expanding, with almost 30 new products in the planning stages. On the way from them at the moment are the new MK5 keyboard and MINI interface (reviewed this issue) and the Afrokit tape for the SpecDrum. They've also told us about a new SpecDrum Users' Group for the 25,000 owners. The group is called Sine and produces a monthly newsletter, on tape, called Sinewave. The group is independent of Cheetah, although the company will be supporting it with news and technical info.

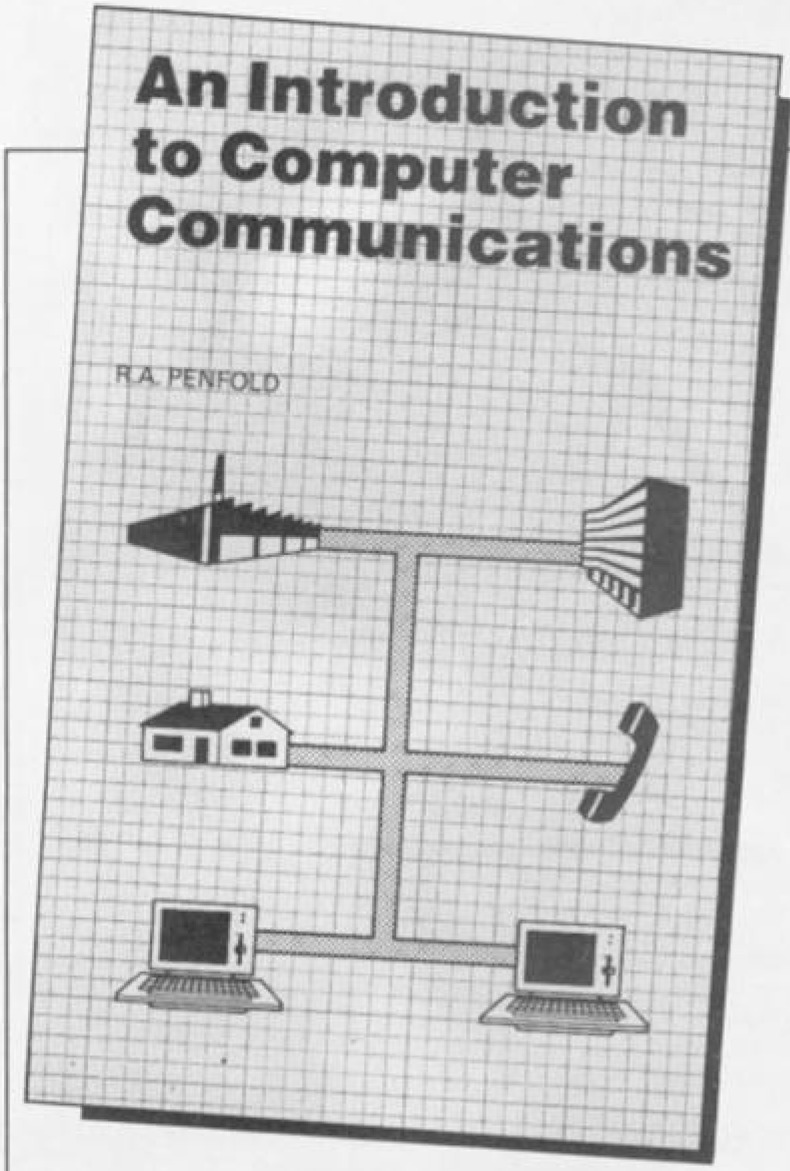
You can contact Sine through M. P. A. Bellamy, 304 Walkley Bank Rd., Sheffield S6 5AR.

Zap happy Hewson

Hewson continue to turn out games almost as fast as we can review them, and their next two releases are on the way soon. Gun runner, due out in May, is a shoot 'em up set on a dying planet where you have to fight your way to supplies which will allow you to save the world. Then in June you'll see Zynaps by Dominic Robinson (who produced the Spectrum version of Uridium). It's another shoot em' up, full of asteroid storms, aliens and space stations. Both games will cost £7.95.



Gunrunner



Comms Book

Computer books have generally been over-priced, but Bernard Babani's range of computer titles is both cheap in useful. Their latest book is an Introduction To Computer Communications, by R.A. Penfold. It's a simple explanation of just whats comms involves, the possibilities and gives a good, simple explanations of the technical side of things for anyone who doesn't understand all the jargon about bauds, parity and such like. At £2.95 it's recommended if you're thinking about buying a modem.

Editor: **Bryan Ralph**
Assistant Editor: **Cliff Joseph**
Consultant Editor: **Ray Elder**
Advertising Manager: **Peter Chandler**

Design: **A.S.P. Art Studio**
A.S.P. Advertising and Editorial
No. 1 Golden Square, London W1R 3AB 01-437-0626

Printed by Chase Web, Estover, Plymouth.
Advertisement Copy Controler: Andy Selwood
Distributed by: Argus Press Sales and Distribution Ltd, 12-18 Paul Street, London EC2A 4JS

ZX Computing Monthly is published on the fourth Friday of each month. Subscription rates cn be obtained from ZX Subscriptions, Infonet, Times House, 179 The Marlowes, Hemel Hempstead, Herts HP51 1BB.

Spectrum Games Top Ten

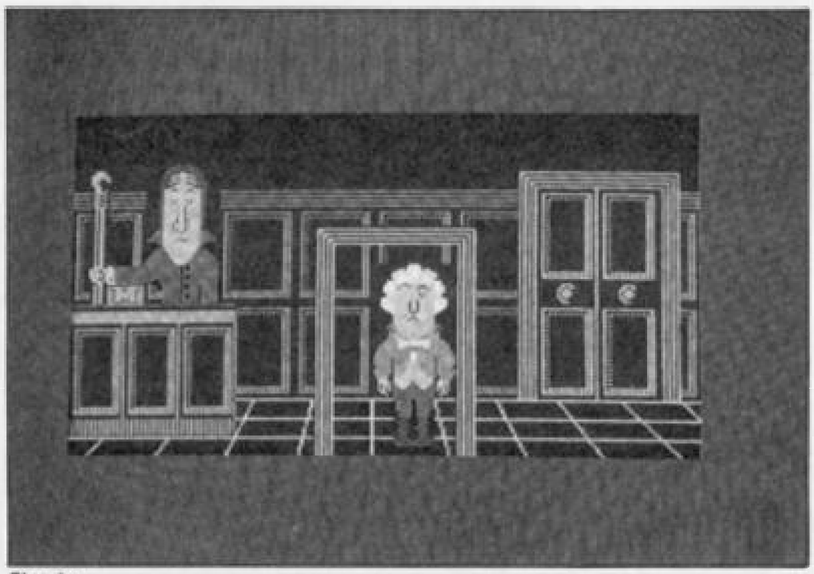
1 ( )	Leaderboard	US Gold
2 ( )	Bombjack II	Elite
3 (1)	Gauntlet	US Gold
4 ( )	Five Computer Hits	Beau Jolly
5 ( )	Konami's Coin Op Hits	Konami
6 (5)	Paperboy	Elite
7 (4)	Super Soccer	Ocean
8 ( )	Dragon's Lair II	Software Projects
9 ( )	Arkanoid	Ocean
10 (10)	Space Harrier	Elite

(Chart supplied by W.H. Smith)

The Big Sleaze

... is the title of Piranha Software's next release. It's a spoof adventure, as we've come to expect from Delta 4 who are writing it, based on 40's Hollywood thrillers.

Also on the way is Flunky, the new game by Trap Door author, Don Priestly. The game is set in Buckingham Palace, where you play the part of a royal flunky rushing back and forth to carry out the royal commands. The games uses the same mega-sprite animation techniques as Trap Door, but this time the playing area has been greatly expanded. Flunky is due for release in May, priced at £9.95.



Flunky

Now! again

Virgin Games have lined up another games compilation for Easter release. Now Games 4 features Dan Dare (good), Mission Omega (so-so), Back To The Future (also so-so), Jonah Barrington's Squash (good, but hard), and Hacker (also good but hard). At £9.95, Now Games 4 should be available round about now.



Spot the ball

Is the picture above
a) The Virgin crew playing spot the ball in order to publicise the updated version of F.A. Cup Football called would you believe F.A. Cup 87.
b) The Virgin crew being rather silly?
Answers on a postcard to Richard Branson, c/o A big boat somewhere in the North Atlantic.



Booming Trivia

Domark's Dominic Wheatley and Marc Strachan are trapped in a sartorial timewarp. The Wacky Duo turned up in 60's regalia to announce the release of Domark's Baby Boomer edition of Trivial Pursuit. All the questions on the tape are to do with the drug-crazed goings on that took place during the 'baby-boom' years of the 60's.



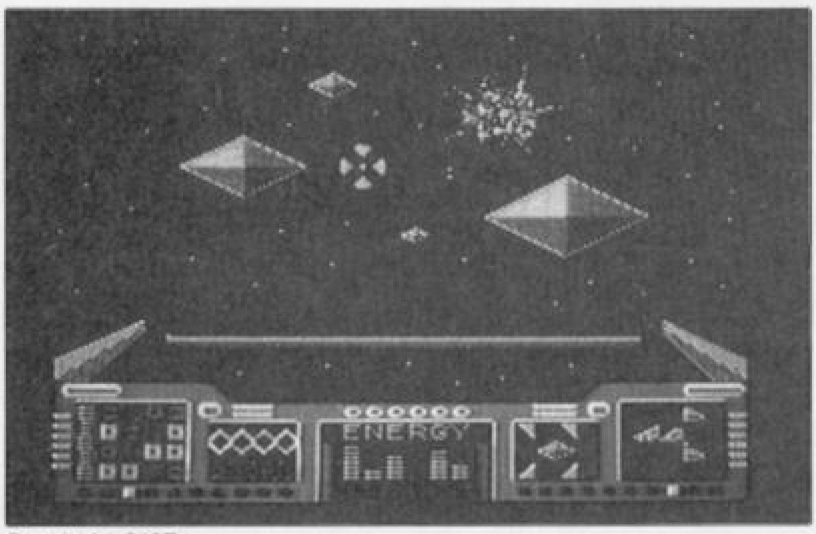
Ugh!

Steve Brown and friends pose in designer loincloths to announce Palace Software's next release, Barbarian. A two-part combat game with a Sword and Sorcery flavour to it, Barbarian sets you the task of learning the skills of swordsmanship and then setting off to rescue the Princess from the clutches of the evil Drax. Palace's games always feature excellent graphics so the threat/promise of a decapitation scene in the game may stir up a controversy.

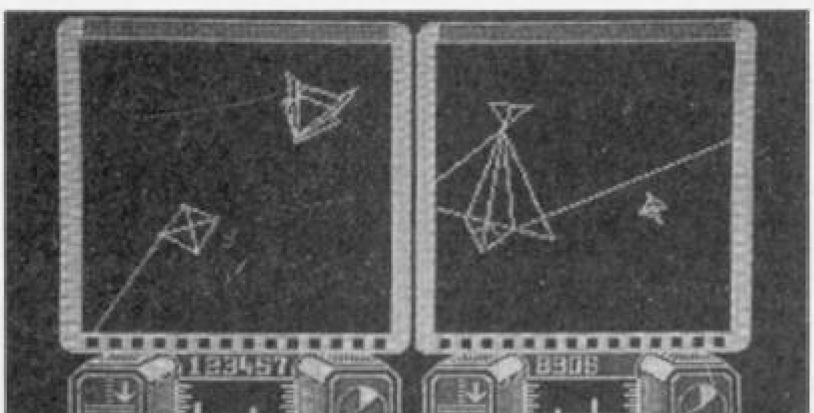


Everything but the kitchen synth

XRI Systems have just released their CZ101 Editor software. For use with their Micon MIDI interface, the software allows you to control a Casio CZ101 synthesizer keyboard. The program comes with 96 pre-programmed sounds as well as allowing you to create sounds of your own. The Editor Software is priced at £22.95 and the Micon Interface at £49.95, and you can contact XRI for more information on 021-382-6048.



Dogfight 2187



Deathscape

Death and Dogs

Starlight Software, the new outfit formed by Francis Lee, follow up Greyfell with two new titles, Dogfight 2187 and Deathscape. Dog fight is a futuristic two-player combat game in which you have to fight off hordes of aliens who are pouring into the universe through a great hole in the space-time continuum. Only the Spatial Generator can heal the rift but it's pieces have long been lost all over the known universe, and it's up to you to find them. Deathscape puts you in charge of a fighter space ship and, umm, you have to fight off the fleet of invading aliens...

Oink!

CRL's future releases include, Frankenstein, Traxxion and Oink! Frankenstein is a three part graphic adventure based on the novel, in which you play Victor Frankenstein, the man who created the monster. Traxxion is a shoot 'em up set within the Tau Ceti milieu, in which you have to prevent a space station from blowing up when its reactor goes into meltdown (sounds like our coffee machine). Then there's Oink! This is a tie in with the magazine of the same name. Unless you've seen the magazine then describing the game is a little bit tricky, but it features characters such as Rubbish Man and Uncle Pigg for starters.



CROSSFIRE

News and views from ZX readers

Cold chips

I was most interested to read the letter from John Wase in the CROSSFIRE column of the March issue regarding the problems he is experiencing with his 128+2 in cold conditions. If it is any consolation to him, I experience the same problem every time I switch on to dash off a quick letter.

I replaced my 48K with the Plus 2 a couple of weeks before Christmas and hooked it up to my Interface 1 and Microdrives and loaded The WRITER which I had used successfully with the 48K, and found to my dismay that CAPS M and N would not register and cursor UP and cursor RIGHT were dead, also STOP was sluggish. Straight away I rushed it back to Dixons in Bristol who replaced it without question, and you've guessed it, same problem. This happened three times, although I admit that on each occasion it was only powered up for about ten minutes while I confirmed that the fault was still there. By the time I got the fourth one Christmas had arrived, and as the shops were closed I was stuck with it. Consequently I discovered that the fault cleared after the warming up period. Why don't they fit a choke to it?!!

Having now got my Plus 2 well and truly warmed up, I may be able to help D.H. Pipe of London NW10, who was enquiring in CROSSWIRES about his Rotronics DX85 printer. I suspect that it was manufactured by SAMLECO COMPUTER SERVICES LTD, SCS House, 9 Fairacres Industrial Estate, Dedworth Road, Windsor, Berks SL4 4LE. Tel. 07535 54717/8/9. I have had a Samleco DX85 for about twelve months now and find it works very well from the instructions in the manual supplied. Also, I believe SAGA marketed the DX85 under their name until recently. If Mr (or maybe it's Ms Pipe) contacts Samleco they may be able to help.

Finally, as a senior citizen, I would like to say how much I enjoy ZX Computing. My particular interests are utilities and technical items (but not too technical!). Keep it up!
J.F. Howell
Bristol

LLIST

Reading my letter in the March issue of ZX ('Printer tips') I would like to add that LLIST only works on short programs. However, compiling an index from the last 12 issues I came across a letter by J. Walters with a M1009 printer who increased the baud rate of his printer to 9600 and got perfect LLIST and COPY. Ignoring the handbook and throwing all caution to the wind I clicked the two vital switches to 'off' (SW-2-4 and SW-2-6) and hey-presto LLIST and COPY work fine on the HR-5 at greatly increased speed and now it is only one cup per copy instead of two or three.

As my Interface 1 had a nervous breakdown I decided to write my own little word processor in 128K mode. And here is my query: on running it runs in 48 mode as indicated by the cursor and will also accept keywords but prints out with the RS232 port and returns back to 128 on returning to the listing. The same happens to the index file I compiled in basic and a hexloader for the 128. I thought you couldn't return to 128 from 48?

About J. Wase's letter "Out in the cold", my 128 will give Tape loading error in the first few minutes with any tape, however the keyboard works correctly from the word go.
K.A. Shimmel
Nottingham

As Toni Baker shows this issue, you can switch between 48 and 128 modes if you want to, so hopefully her new series will answer any queries you've got.

Wither Kempston?

Could you please send me information and dealers' addresses for the 128/Plus 2 compatible joystick interface, as reviewed in December's ZX Computing.
Michael Halb
Sheerness.

As Kempston's range of joystick interfaces are more or less accepted as setting the standard for the Spectrum you should be able to find them in any decent computer shop. Alternatively you can contact Kempston Electronics direct, by 'phoning 0908-690018.

Back issues

I have been unable to obtain your magazine ZX Computing since July 1986 and wondered if it had gone out of print until I found the February 1987 issue on a visit to Glasgow recently.

As a result I have missed the articles on the Opus Discovery column and also utility programs for the Spectrum 48K. I wonder if you could let my have reprints of these articles or suggest how I could obtain them.
Dr. R. Shepherd
Glasgow.

Just as a general point, anyone wishing to obtain back issues of ZX can do so by contacting ZX Computing Back Issues, Infonet Ltd, Times House, 179 The Marlowes, Hemel Hempstead, Herts HP1 1BB.

Address manager

The main purpose of this letter is that when looking through some back issues of ZXC I noticed in the Dec. 86 "Crossfire" on page 88 that a Mr. Tyson of Chester was in trouble with the OCP Address Manager with the Opus disc drive. All the problems have been solved and if he cares to contact Mr. F.G. Miles, "Windrush", Rabley Heath, Welwyn, Herts AL6 9UF he will be willing either to supply "Word Manager4" + "Address manager2" + "Mail Manager" all on a 3½" disc for the Discovery, or to supply the address of EEC Ltd who have taken over many of the original OCP products. The price is extremely competitive and the programs all work very well.
J. Burke
Barnet.

Music mix

I am writing once again to compliment you on your magazine, especially the March issue which had the sound sampler program.

For the past few years I have played guitar and recently done some recording (four track) and would like to know if it is possible to have a program for a mixer to enable vocals or instrument to be mixed and routed in separate channels, solely by the computer or even effects echo, reverberation etc.
Ian Barrass
Hull

Rotronics help

I own a Rotronics Wafadrive and am having difficulty in obtaining more wafas, and several information sheets now that the company has gone into liquidation. I was hoping that you or your readers might know where I could obtain new wafas and information sheets nos. 6,9,11,12,15-21 (or photocopies of these).
Simon Duff
4 Victoria Park
Cambridge CB4 3EL

We know that there are a lot of Wafadrive users amongst ZX readers so perhaps someone can help out by contacting Simon at the above address.

'81 in the USA

I read your article in the February 1987 issue of ZX Computing and am responding to your "guestimate" that "As far as we can tell there are no longer any companies producing ZX81 equipment. . ."". I know a number of USA repair facilities and Sinclair suppliers. One supplier in particular, Tom Wodds, produces many fine peripherals etc for the ZX81, including a universal I/O port for any Sinclair Z80A (except ZX80). He also has an extended BASIC program and MC debugger. Many suppliers still support and improve the ZX81 (called TS1000 here).

I belong to a very nice User Group here in the Chicago area and would be very happy to help any of your readers find ZX81 sources.
Edwin L. Schoen
3939 Glendenning Road
Downer's Grove, Illinois
USA 60515

and COMPUTER SPARES

DISCOUNT JOYSTICKS

...SS ... STOP PRESS ... STOP PRESS ...

BUDGET SOFTWARE

PRESS

SPARE PARTS

A NEW EXCITING OFFER FROM THE VideoVault



The Videovault will shortly be releasing the **most up to date and exciting** catalogue ever produced, our team of experts have scoured the globe to find you all the latest software and add on's for your computer, **all the latest** in joystick technology, all the top and tested **software** titles all at **great discounts** to you. Anyone can apply now using the coupon below and enclosing 2×18p stamps for your **free copy**. We have all the top branded items available, delivery by **return post** (our usual standard). All goods carry our guarantee of satisfaction. Never has such a catalogue ever been produced before.

We will be updating the products a soon as we have all the latest news available to us, on all the latest technology.

SPECIAL EXCITING OFFERS ON EACH ISSUE

Top reviews, a 1st again for the **Videovault**, **the No.1 computer company in the UK**, (we must be as all our competitors keep copying us.)

SHOP UNDER ONE ROOF FOR ALL THE TOP PRODUCTS AVAILABLE

A SAMPLE OF OUR FANTASTIC DISCOUNT PRICES



DISCOUNT KEYBOARDS

The Saga Lo Profile Professional Keyboard

Recommended retail price **£39.95**

Videovault discount price only £28.50 + £1.50 post + packing, total £30

Computer repair and upgrade to Saga £50.00 complete.

The Saga Lo Profile Keyboard has 52 full travel keys, featuring a separate numeric keypad and printed keys. Your Spectrum fits securely inside the housing.

Dimensions: 22×44×5cm. Suitable for Spectrum 16/48K and Spectrum Plus.

OVERSEAS ORDERS MOST WELCOME

P.S. SPECIAL BONUS OFFER FREE

the 1st 500 customers ordering from our new catalogue will receive free of charge a quartz digital watch. No minimum order value required.



PLEASE RUSH ME YOUR EXCITING NEW CATALOGUE AS SOON AS IT IS AVAILABLE HOT OFF THE PRESS.

I ENCLOSE 2×18p STAMPS TO COVER POSTAGE COSTS

MR/MRS/MISS ______________________

ADDRESS ______________________

____

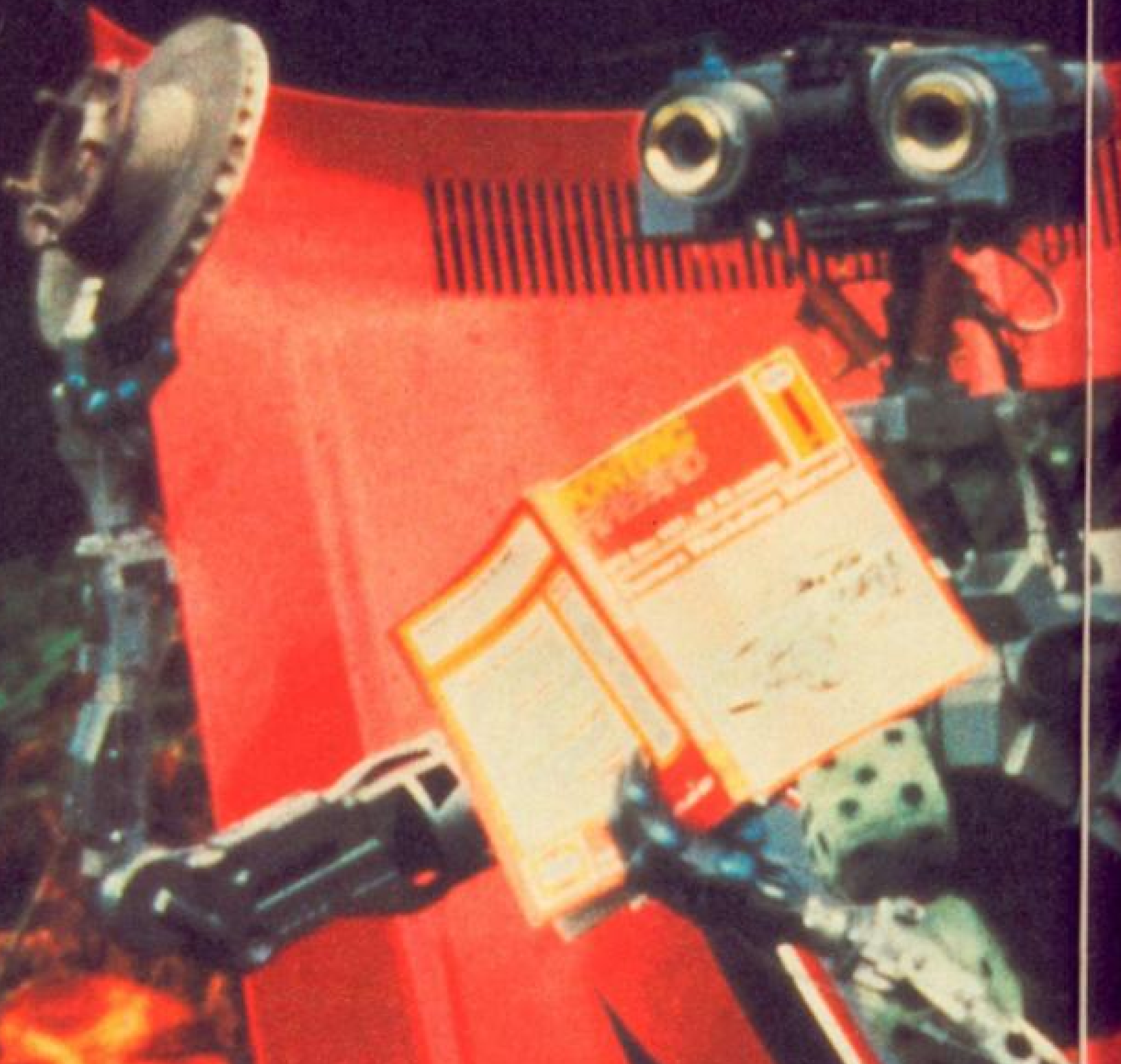
____

______________________ POST CODE ______________

CUT OUT THIS COUPON AND POST IT TO: BLOCK CAPITALS PLEASE

VIDEOVAULT LTD, CATALOGUE DEPT, 140 HIGH STREET WEST, GLOSSOP, DERBYSHIRE, SK13 8HJ ENGLAND

A robot with soul in Ocean's arcade adventure.



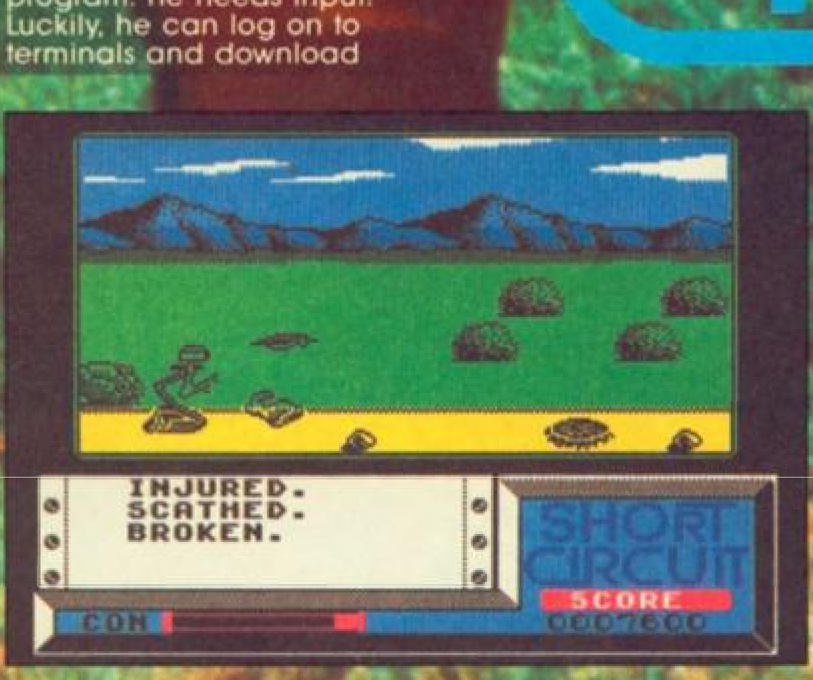
Short Circuit
Ocean
£7.95

SHORT CIRCUIT

"Number five — Alive!" is the plaintive plea of a robot that was hit by lightning in a freak thunderstorm and is now alive! You must help him escape from Nova Robotics and keep him alive! The only trouble is that he's a SAINT robot (Strategic Artificially Intelligent Nuclear Transport) and he's still armed! The scientist who built him wants to take him apart to find out what went wrong, Nova's boss wants to capture him before millions are killed and the security chief wants to blast so he can get home in time for dinner. You must keep him alive.

Based on the superb film, Short Circuit (highly recommended — go and see it immediately!) the game is split into two parts. In part one you must escape from the Nova Robotics complex with the essential jump and laser accessories and in part two you must flee for your life and avoid the pursuing guards that aim to disassemble you! You can attempt the parts in either order but it's logical to begin with part one.

The game begins with Number 5 alone in a room full of mysterious desks, chairs and plants and he's only got a LINK program. He needs input! Luckily, he can log on to terminals and download additional software. After a little exploration you'll soon find a terminal and be able to download the programs to SEARCH, USE and READ. Now you can explore the rooms of the complex, SEARCH for useful objects which you can then USE to solve the games problems. Searching couldn't be easier as all you do is stand in front of an object, select the SEARCH program on a duckshoot menu and press the fire button. The results of your search will be reported on your printer, which also presents you with single key options if you've found an object you can pick up or books to





read ("ooohhh input!") that provide important clues.

As you progress through the complex you'll find locked doors, file cabinets and units that require keys and coloured passes to open them and eventually another terminal. Now you can download three more programs that allow Number 5 to DROP objects (useful as he can carry only three at a time), BREAK and LOOK with which you can identify objects. For example, something that you thought was an empty desk actually turns into a fruit machine and a use for the pound coin that you found stashed away in a plant pot.

Mapping

As with all arcade adventures a map is essential particularly if you mark on it where you found each object, no matter how obscure, as might find a use for it although it's hard to imagine why you'd want a coffee cup, paper clip or scared mouse.

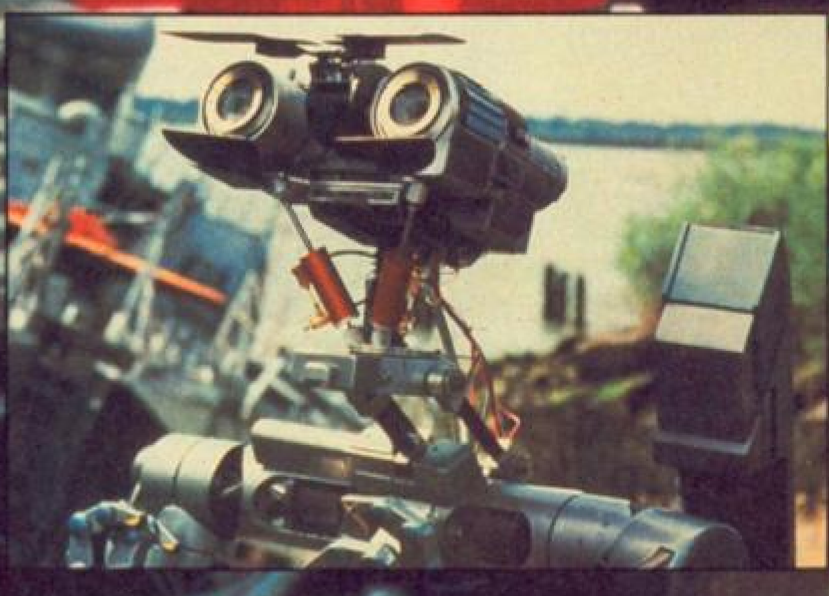
In part two Number 5 is literally on the run as he's being chased by other SAINT robots and he's determined to get away. Armed with the laser and jump mechanisms, you must hurtle along a country road strewn with rocks and wildlife.

By now Number 5 has developed a conscience and must avoid disassembling the birds and crawling creatures otherwise he'll get so depressed that he gives up.

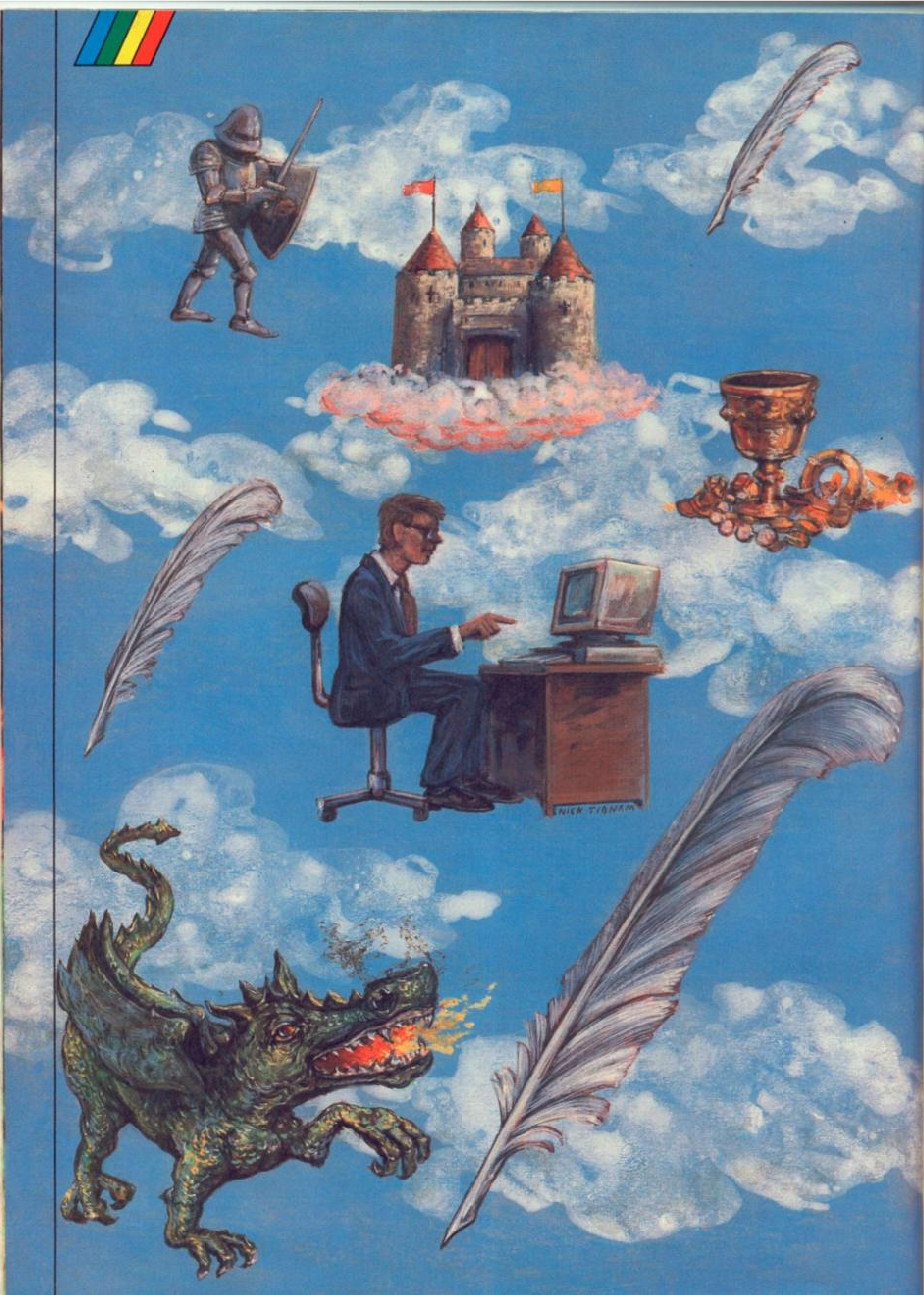
Using keyboard or joystick controls you must time your leaps and ducks to avoid the wildlife while also avoiding the boulders that will injure and break him while keeping a finger near the laser fire button in case a robot or guard gets too close.

Number 5 is alive! It's up to you to keep him that way.





MONSTER HIT



P.A.W.S.

After the release of G.A.C., Gilsoft strike back with the Professional Adventure Writing System.

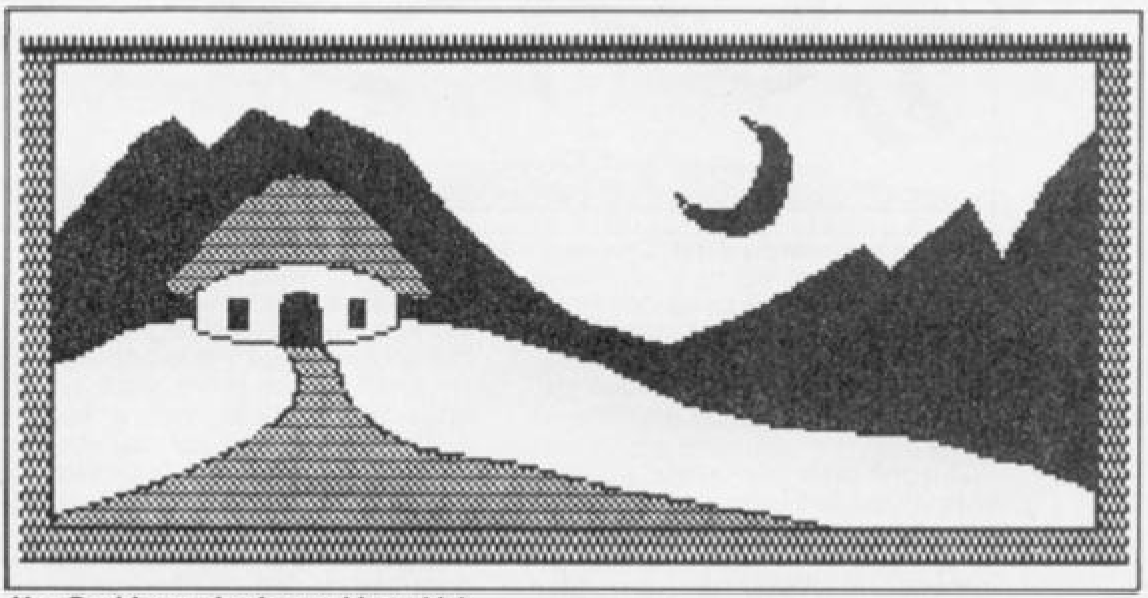
P.A.W.S.
Gilsoft
£22.95

The release of Incentive's *Graphic Adventure Creator (GAC)* caused great excitement last year — to be followed by much debate in adventuring circles about whether it really did represent a significant advance on Gilsoft's long established Quill. Suddenly, however, it seems largely irrelevant whether it did or not. The reason? Well, coming up on the port bow, and preparing to deliver a broadside which looks as though it will settle the matter for good, is *The Professional Adventure Writing System (PAWS)* from Gilsoft.

You might be excused for doubting that there really is a need for yet another adventure-writing system. After all, you might think that with GAC and the old Quill (with its various add-ons) to choose from, surely any budding adventure author worth his salt should be able to cope. If you have such doubts, let me dispel them at once. Gilsoft have been at this business a long time, and in PAW their experience shows. This is a magnificent piece of software, by any standards, and it's in a league of its own.

The system

For your £22.95 you get a cassette and two manuals in a large box. The first of the manuals is a 66 page introduction which gets you started by leading you through the creation of a short adventure. The second one is a technical manual which wasn't complete at the time of this review, though most of the essential information was provided.



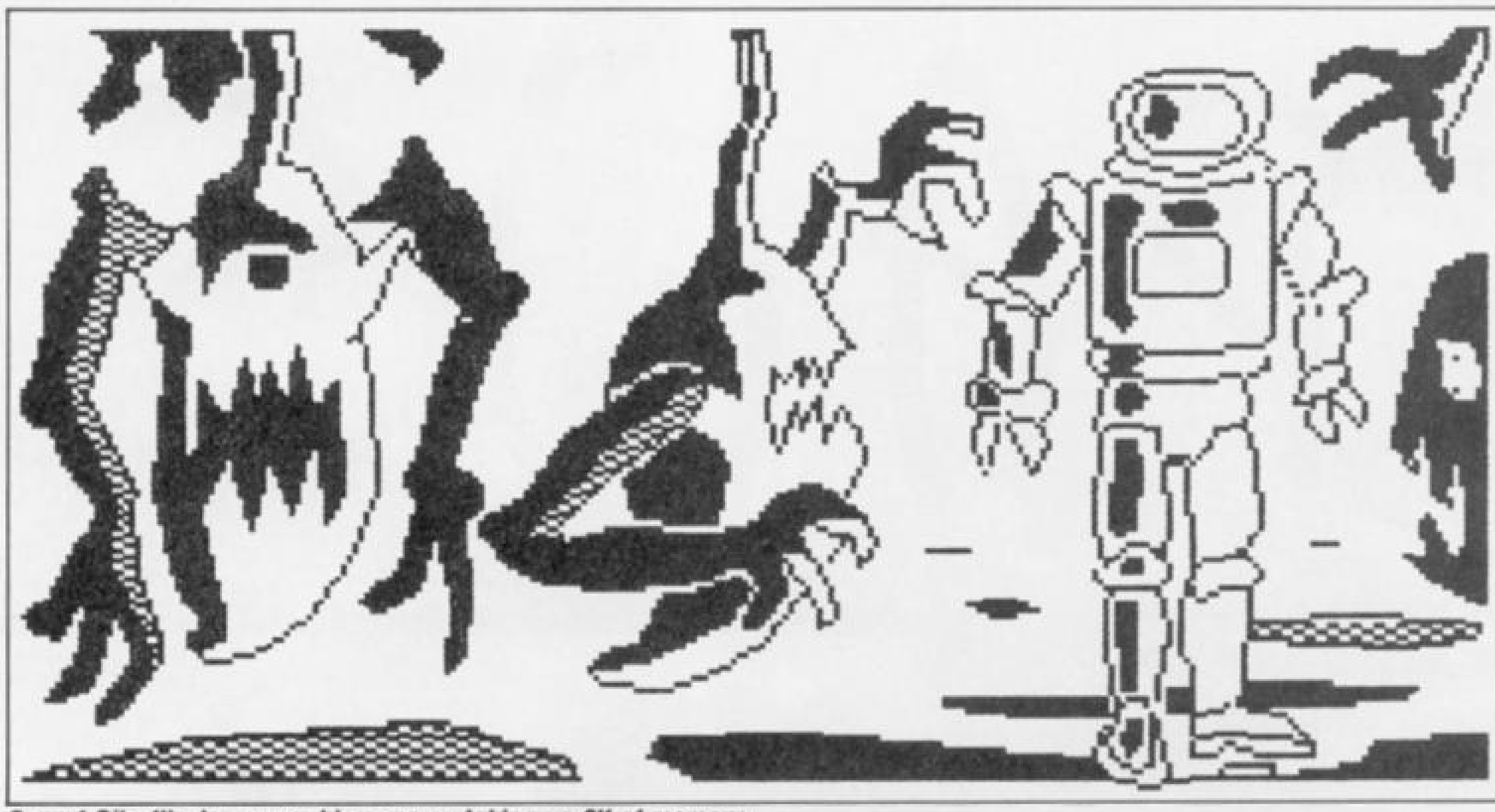
Alan Davis' own simple graphics, which takes up just 500 bytes (½K).

In order to release as much memory as possible for your adventure, PAWS uses a system of 'overlays'. These are recorded on the cassette as a series of short code blocks following the main program. Most of these are loaded in straight away so that at start-up most of the program's extensive facilities are available. Then, as your adventure database grows in the writing, these sections are overwritten, but can be loaded in again separately as required. This isn't much of a problem for the 128K Spectrum owner, nor will it be a problem with the microdrive version that is promised — but for those of us with 48K machines a good deal of to-ing and fro-ing with the casette recorder is going to be necessary. There are, I think, two views one can take of this. The first is that the program is so good that it almost justifies the purchase of a 128K machine anyway; and the second is that the program is so good that the hassle of fiddling with the tape is a small price to pay for the privilege of using it!

Completing the roundup a short adventure database is provided at the end of the tape, as an example. This can be loaded into the system and hacked apart at will. And the whole of the other side of the tape contains — wait for it — 22 (yes, twenty two) different character sets! This may seem like gilding the lily, and yet it is typical of the whole style of presentation which underlies PAWS. Gilsoft have done everything in their power to help you to avoid that stereotyped appearance which has been a feature of so many Quilled games in the past.

Menus

Once you get the program loaded in and running, everything is menu-driven (no icons, thank goodness!). The overall 'feel' of the program will be quite familiar to those who've had experience of the Quill, though once you actually get involved, you begin to realise the enormous potential of it. It's easy to get started, using the introductory manual as a step-by-step guide, but make no mistake — you won't master all the intricacies of PAW for some time. I must emphasise that this



One of Gilsoft's demo graphic screens, taking up 2K of memory.

doesn't mean the program is difficult to use. Rather, it means that you'll go on and on discovering new possibilities as you gain experience with the system.

I can't possibly cover everything that the program is capable of here, and it would take ages to explore the full potential of PAWS, but I can give you some idea of its scope by selecting a few examples. The parser, for instance, is probably as sophisticated as anyone will ever need. It accepts multiple commands (using punctuation marks, AND, or THEN, as phrase separators). ALL and IT are catered for — so that an input such as "OPEN THE CHEST, LOOK INSIDE IT, AND GET EVERYTHING OUT OF THE CHEST" can be dealt with in a straightforward manner. The system anticipates that you may want to have other characters in the adventure, so speech can be accommodated using inverted commas in good old "Hobbit" fashion.

Language and logic

When it comes to sorting out the game logic, I can only say that the Gilsoft chaps seem to have anticipated pretty well anything you could possibly want to do. Again, I can give you no more than a taste, but you can, for example, define certain objects as 'containers' (i.e. capable of containing other objects) — and this alone can give rise to enormous fun, as you'll see a little later on. 'Flags' are needed to keep track of event (such as whether doors are open or closed) in an adventure, and there are 256 of these, but not all of them are available for use (though they can all be inspected) since the system uses some of them itself (eg. to store the player's current location). That still leaves more than enough for you to work with.

The 'language' you use to program your game is similar to that used in The Quill, but extended in scope almost beyond recognition. Basically, commands are divided into conditions ("if so-and-so") and actions ("Do such-and-such") — but the list of available commands fills two pages of the manual! Possibly the best way to illustrate the flexibility of the system is to describe a particular sequence of events that took place while I was away actually working through the examples suggested in the introductory manual.

In the bag

I'd arrived at a stage where I had a partly working small adventure, with a few locations in a park and a few objects. These included a canvas bag, defined as a 'container', and inputs like "PUT EVERYTHING IN THE BAG", "LOOK IN THE BAG", and so on, were working beautifully. At this point I put the manual aside and thought what a clever trick it would be to allow the player to get inside the bag himself. So the canvas bag became "an enormous canvas bag", the inside of the bag was defined as a new location, and the necessary commands added to the 'Response Table'. Next I wondered if we couldn't have some other character come along and pick up the bag while the player was still in it (and could this be done in real time?)

Well, it could indeed be done. All of it. There's a facility which enables the adventure program to break away from player input after a specified time and scan the 'Process Table' (which holds a set of condition tests and actions which don't depend on input from the player) so that real-time independently acting characters are perfectly possible. By manipulating a couple of flags to keep track of what was happening, I found myself being merrily carried around the park inside the bag, by someone else, free to hop out or back in again at any point.

Although this was just a bit of fun, fans of The Hobbit will remember certain notorious escapades with a barrel and a butler... Well folks, you can do all that with PAW, and it is very impressive indeed!

48K owners will possibly be worried about available memory, especially since this was an area where the GAC was somewhat deficient. At start-up, the 'Free memory' option tells you that 25434 bytes are available for your game. This might not seem significantly more than the 23194 that GAC offers, but in fact it is — for several reasons. PAWS already contains a large vacabulary and a variety of messages; since they're already there, you won't have to use precious memory to put them in yourself. The program also possesses a text compression facility which works superbly. I tried this with 1500 bytes of adventure on board, and was rewarded by the system announcing that it had just saved me 600 bytes. This is very good news indeed, and suggests that the nominal 25K of free memory will effectively be

the equivalent of considerably more than 30K, taking compression into account.

Great graphics

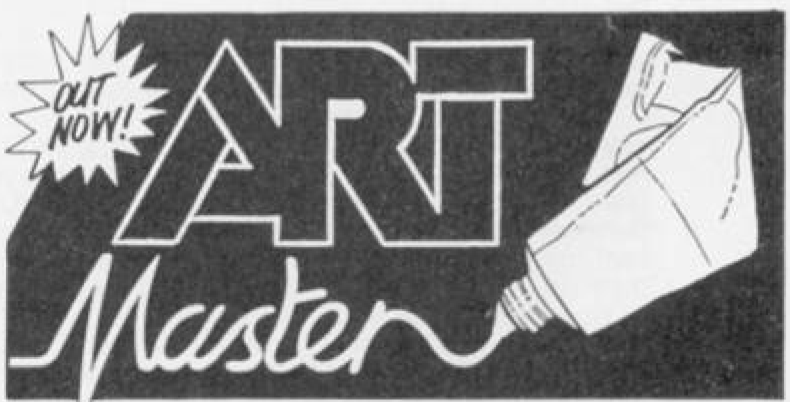
So much for game construction and logic. What about general presentation and graphics? As far as text is concerned, printing is automatically formatted onto the screen — with a word-wrapping system that works properly, unlike GAC's which doesn't, quite. Objects present at a location can be listed vertically, with each object on a new line, or as a piece of continuous text with the objects neatly linked by commas and "and" inserted where necessary. The choice is up to you. Several character sets can be stored at once, and the program will swap between them if required. In short, your text can be as varied and interesting in appearance as you care to make it.

In order to design graphics, as illustrations to particular locations, it's necessary to load in an overlay. As a bonus, this includes a character designer which can be used to change the shape of letters (in case none of the 22 sets appeals to you...) and also the 15 patterns used for shading in the main graphics routines. The pictures themselves can be any size you like, and you can arrange for them to scroll away as text is printed — or to stay put while the text scrolls below them.

The method for designing the pictures is effective and easily mastered. Points can be plotted, and lines drawn, areas can be filled either with solid ink, or a chosen pattern of shading, and areas of colour can be blocked in separately. Small pictures can be designed and called as subroutines for inclusion in other pictures at any one or eight scales of magnification. I found some difficulty in positioning these 'subroutines' correctly within larger pictures, but it's early days yet, and I'm sure things will improve with practice.

There's no escaping the fact that graphics tend to gobble memory. PAWS' system, like GAC's, stores pictures as a string of graphic commands (rather than as a chunk of screen memory), which would be much less economical). The facility to call graphic subroutines also helps to save on bytes. For instance, if you're willing to use a standard picture of a tree in several locations you save memory because the sequence of commands for drawing the tree need only be stored once. Still, some of the pictures in the Gilsoft demonstration adventure use more than 2K each, for all that. They're very attractive as advertising material, but it would hardly be practical to devote so much memory to individual pictures (on a 48K machine, at least). Scattered about on these pages you should find some screen dumps to illustrate this. The good, highly detailed ones are the Gilsoft 2K jobs; my own rough doodles are much more primitive, costing only 500 bytes each, so you'll be able to pick them out with little difficulty! They won't please your eye much, but they should give you a better idea of the amount of detail you can actually afford to include in a 48K machine.

To sum up then, PAWS is precisely what its name claims it to be: a professional adventure writing system, with the emphasis (as far as the *potential* quality of its product is concerned) on "professional". It's the kind of program that an adventure writer dreams about, with what seems to be virtually unlimited scope — and I can say this confidently without even having seen the full details of the technical manual. It deserves to sell like hot cakes, and if you're at all interested in adventures, it's an essential purchase. Bring on the microdrive version, please...





Cheetah follow up the SpecDrum with some more musical add-ons

At £99 the MK5 keyboard is the cheapest of its kind on the market, and in order to make it inexpensive several features have had to be sacrificed.

The keyboard is a set of plastic, piano type full sized keys covering five octaves (61 keys). It has a pitch bend wheel, a multi function single click key, four indicator LEDs and a three digit display panel.

The keyboard itself does NOT produce any sounds but sends signals in MIDI code out of its single MIDI OUT socket on the back, there is one other socket and this is for the 9V power supply. The idea of this keyboard is that it controls a range of other devices, a synthesizer, a synth module (like a synthesizer but without a keyboard) or a sound unit via MIDI interface (such as Cheetah's own MINI interface) linked to a computer.

Unlike many modern keyboards this one's care is made of metal which is dark grey and solid. The keys themselves have a very positive feel and give the impression that they will last a long time.

I am not a great fan of the increasing use of the 9V mains adaptor plug connected by a typical Spectrum plug to supply power, but it works OK and didn't give any problems, though trying to plug several of these large plugs into a socket board can cause problems at times.

The MIDI socket is the standard five pin DIN unlike the odd type fitted on the 128/+2. Through this socket you can transmit the obviously necessary note on and off codes and, as you may have surmised, pitch bend data. The channel that it is transmitted on can be selected, a program change and the octave range can also be specified and sent.

Operation

At first the means of operation seems rather complex. Even though there is a single button to control all the functions this is used in conjunction with seven of the keys themselves. These are the top seven and are marked above each key with the function it controls — Channel, Octave and Program. Two keys are assigned to each, Up and Down, and the final key is the Transmit command.

The four LEDs are marked from Channel, Octave, Program and Program Mode. When simply playing then the keyboard is in Program mode and that LED is lit.

To use one of the other features the Program/Play switch is clicked and the required up/down key for the function you want is pressed. The corresponding LED lights and the display panel cycles up or down through the numbers in the range of that function — 1 to 16 Channels, -1,0,1 for Octave and 1 to 128 for Programs. Once the desired number is displayed then pressing the transmit key sends that instruction to the device in use.

An interesting decision was to fit the pitch bend wheel on the top of the keyboard with a left/right action rather than the more usual side location with up/down action. I suspect that this was dictated by the case design rather than choice, but it actually works well and once I got used to it I prefered it to the customary system.

Obviously the keyboard had to lose certain features to be produced so inexpensively and whether these make this keyboard viable or not depends on your intended use for it.

If you have a MIDI equipped keyboard with mini keys and less than five octaves then this is an ideal way of expanding it. I used it with my Casio CZ101 and it was like playing a different machine, the playing "feel" was much improved and the sound range enhanced with some really deep bass or very high top notes. It was a perfect match.

However, the keyboard has not got 'velocity' or 'after touch'. The first provides a means of altering the sound depending on how hard/fast you hit the keys and the second allows variation in a sound depending on the pressure with which you hold the keys down. The second tends to be implemented only on very expensive machines, but most sound modules provide for velocity and it is very useful for expressive playing.

Used with some computer software this need not be a disadvantage, the XRI Micon interface and Step Time program allows for individual setting and editing of velocity. So if its prime use is with this, or similar, then it is not a real disadvantage.

The system of sending program and channel changes takes several seconds, and this might be a disadvantage in live performances, usually it is desirable to alter a sound in mid-song by a quick, single key press. This is not possible with the MK5.

Though limited by fiddly function change operation and lack of velocity I am impressed by this product and can see it being a valuable addition for many people's systems.

MINI Interface

If you haven't got a synth, or synth module, and you want to play music but don't want to spend a further £150+ on top of the MK5 then your answer could be the MINI interface from Cheetah at £29.95. This plug in upright box and software for the 128/+2 will allow the MK5 to use the 128/+2 sound chip to play three note polyphonic music.

In addition, the interface's built-in software provides a key split option (where the lower part of the keyboard can play a different sound to the upper part), facilities to edit sounds and create your own, adjust pitch and add or control a tremolo effect.

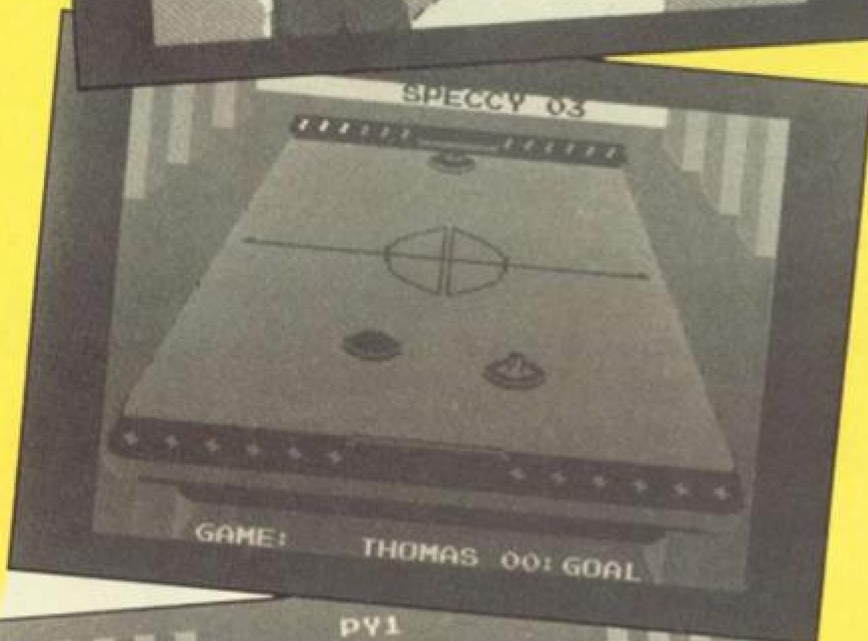
The unit is certainly not up to professional standards, but as a cheap alternative to real synthesis or as an introduction to it or as an improvement over the standard sound features of the Spectrum then it may be worth your consideration.

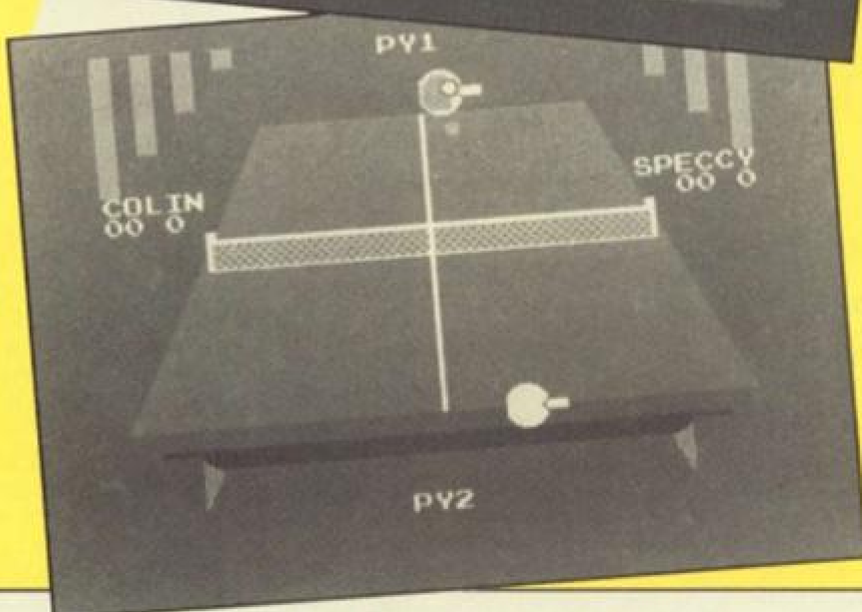
Get on target!

COMPETITION

Your chance to win a top quality dartboard and darts set in our Indoor Sports competition







Advance Software's last sports simulation was Hardball, now they've forsaken the great outdoors to produce a great indoor sports compilation of four games on one tape. Three of the games, darts, ten pin bowling and table tennis are familiar but the game likely to attract most attention is "air hockey". This craze is currently sweeping America and demands great reflexes from the players. It's a tabletop game played on a glass smooth surface and the aim is to slam your puck into your opponent's goal. The simulation has variable playing speeds — fast, very fast and impossible.

To enable you to judge just how close Advance have got to simulating darts we are offering three sets of top notch Dartboard and darts to the winners of this month's competition.

All you have to do is answer the three questions below and write your answers on the coupon provided.

The Questions

Which indoor games are these terms usually associated with?

1. A smash
2. A Spare
3. An oche

Send your entries to Indoor Sports Competition, ZX Computing Monthly, No 1 Golden Square, London W1R 3AB.

The competition is open to all ZX readers except employees of Argus Specialist Publications, Chase Webb and Advance Software.

The editor's decision is final and no correspondence can be entered into. Please remember to write your answers on the back of the envelope.

The closing date is: 5th June 1987.

INDOOR SPORTS COMPETITION

1 ....

2 ....

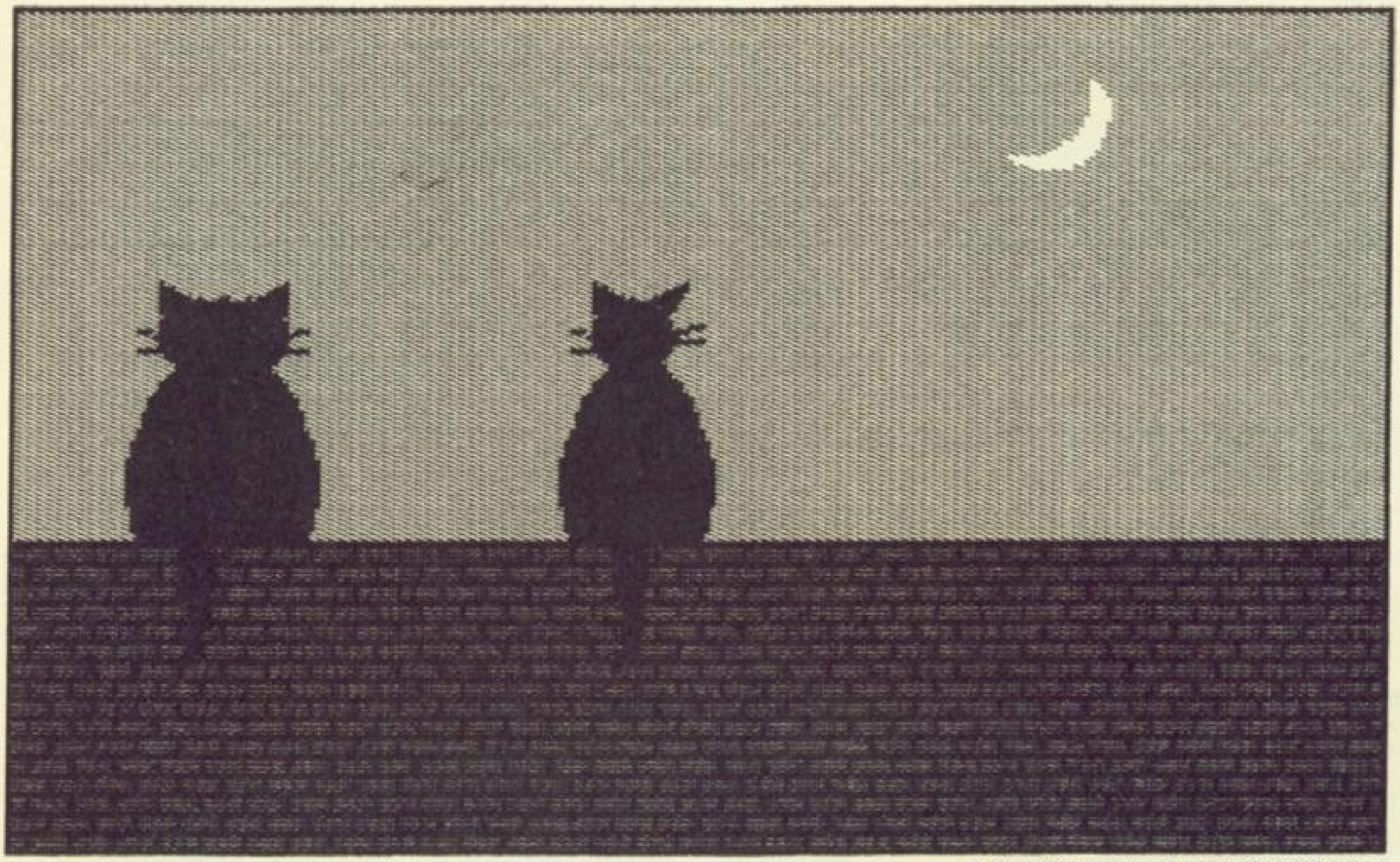
3 ....

Name: ....

Address: ....

....

Send your entry to Indoor Sports Competition, ZX Computing Monthly, No 1 Golden Square, London W1R 3AB



Drawn with the Art Master using 176 bytes.

ART MASTER

Carol Brooksbank looks at a new graphics utility.

Art Master
Summit Software
£9 95

This new screen art program for the Spectrum is a rather unusual mixture of graphics package, extended BASIC and compiler. When the program loads, the screen is blank except for a white asterisk, showing that you are in the mode which offers a choice of ten options, all accessed by pressing the appropriate letter key. A menu would have been a nice touch here, saving the constant reference back to the handbook to discover which keys are which.

The option to choose if you want to draw a picture is the Text Editor, which appears to return you to BASIC. However, it is not the ordinary Spectrum BASIC, but a special one which offers three extra commands — auto line numbering, renumber and edit. The BASIC commands for drawing your picture have to be entered as REM statements, and the auto line numbering gives you line numbers in your chosen step followed by REM. Some of the commands are very similar to ordinary BASIC ones — plot, draw, circle, border — but there are also also new ones such as box, triangle, arc, window, fill copy (copies the window to another part of the screen) and flip1 and flip2 (mirror the window vertically or horizontally). Most of these commands are followed by co-ordinates, either pixel or character square.

Commands can be written out in full, or abbreviated to the shortest number of letters which will identify them. (P for plot, but CI and CO for circle and copy). Colour is handled by the ATTR command, followed by four numbers, for ink, paper, bright and flash. MASK, followed by two numbers, handles over and inverse. There are ten fill patterns supplied, and you can redefine them, or add more if you wish. CALL and JUMP instructions, similar in operation to GO SUB and GO TO, are available, but you have the choice of using labels or line numbers.

Memory saving

Writing the program to draw a screen is very similar to using the ordinary BASIC commands, but when it is complete, you return to the main program and select the compiler option. You are prompted for a start address, and this will be the address to call to draw your picture on screen. Syntax is not checked while the program is being written, so syntax errors stop the compiling with an error message 'Mistake in line ...' When the compiling is complete, the final address used is given. This is where you see the program's great advantage. The compiled code for the cat

Listing

```
 10 REM attr 0,2.1,0
 20 REM window 0,14,31,7
 30 REM clw
 40 REM plot 0,63
 50 REM draw 0,0
 60 REM draw 255,0
 70 REM draw 255,63
 80 REM attr 0,1,1,0
 90 REM window 0,0,31,13
100 REM clw
110 REM plot 0,64
120 REM draw 255,64
130 REM draw 255,175
140 REM draw 0,175
150 REM draw 0,64
160 REM plot 104,64
170 REM arc 112,102,-40
180 REM plot 128,64
190 REM arc 120,102,40
200 REM circle 115,107,8
210 REM plot 108,118
220 REM triangle 108,112,114,114
230 REM plot 122,112
240 REM triangle 125,118,118,114
250 REM plot 123,106
260 REM draw 128,106
270 REM plot 123,108
280 REM draw 128,109
290 REM plot 104,104
300 REM draw 110,104
310 REM plot 104,108
320 REM draw 110,107
330 REM fill 111,114,9
340 REM fill 121,114,9
350 REM fill 115,108,9
360 REM fill 112,80,9
370 REM attr 0,2,1,0
380 REM plot 112,64
390 REM arc 114,48,25
400 REM arc 110,34,-30
410 REM plot 118,64
420 REM arc 117,64,35
430 REM arc 110,34,-35
440 REM fill 114,60,9
450 REM fill 112,8,6
460 REM copy 12,6,6,13,2,6
470 REM flip2 2,6,6,8,2,6
480 REM attr 6,1,1,0
490 REM plot 185,144
500 REM arc 200,160,80
510 REM plot 185,144
520 REM arc 200,160,120
530 REM fill 197,147,9
540 REM e
```

Listing for Cats picture written out in full.

picture uses just 176 bytes. You can store more than 5 screens in 1K of memory. This is far more than you could possibly store using any other screen compressor I have come across.

But of course, there is a price to pay. The compiled code — Gcode — runs much faster than ordinary BASIC, but it is still BASIC. It takes 16 seconds for the cat screen to be drawn, and you can see each line and curve appearing. But still, careful use of windows, and minimal use of shape filling can make the drawing time acceptable and it is even possible, by flipping and copying windows, to do some rudimentary screen movement. The Gcode can be relocated or modified, and there is a test mode which executes each command separately and gives the address trace. The code can also be decompiled, when you find that your listing has been reduced to the shortest abbreviations, renumbered to give steps of 5, and all labels have been removed and replaced by absolute addresses.

There is no problem about compatibility with storage systems or printer interfaces because all saving, loading, screen dumps, etc. are done from BASIC. The only snag is that you have to make sure you are in ordinary BASIC, not the Editor variety. The handbook gives a simple test, but it is very easy to forget which one you are in, and attempting to save, load or copy, initialise your interface or even call the code to run your picture, from Editor BASIC will crash the program.

This program is not going to make you throw away your copies of Art Studio or Artist II. It feels a rather primitive way of making screens while you are using it, especially because you need to plan your design on graph paper beforehand to work out all the co-ordinates involved. But it is a very handy program to have up your sleeve when you want to use a lot of screens and memory space is tight.

EXPERT SYSTEMS

David Nowotnik on how the 'expert' Spectrum or QL can deal with uncertainty.

So far in this series, we've been pretty sure of ourselves! The expert systems we have used up until now have all required yes/no replies to questions raised by our computer expert. There has been no room for uncertainty in our responses, nor has there been any uncertainty in the computer's answer. This is how everything might be in an ideal world. But life isn't quite like that.

Any expert will not know all the answers to problems in his area of expertise. And, usually, we cannot supply our expert with all the pieces of information which he (or she) requires to make a judgement with 100 per cent certainty. In our dialogue with an expert, phrases such as "I don't know", "I think so", and "Probably not" have to be permitted and then used by the expert in forming his eventual judgement. Even if we can reply with absolute certainty to all the expert's questions, the nature of the problem may still prevent the expert being absolutely sure about the answer.

Added to all these problems, an expert may be faced with contradictory information, and will have to make some assessment to obtain the true answer. An example of that was horribly real for me just recently.

Having volunteered for a police identification parade, I had the embarrassment of being picked out by one witness. When the witness was asked 'Are you sure?', he replied 'I'm quite certain'. If the expert in this situation, the police officer, had relied upon that reply, I'd be behind bars now! Fortunately, he placed far greater weight to the evidence (which could not be proven absolutely) that I was at my place of work when the offence was committed. He used his expertise to make a judgement between two conflicting 'certain' pieces of information.

Subjective decisions

It is these kinds of judgement that a computer expert system will be expected to produce. Some judgements will be subjective, others will apply probability theory and statistics to work out the chance for various outcomes to occur. The subjective part approach will be demonstrated in this, the fourth part of my series on expert systems, and a start will be made on probability theory, which will be applied to expert systems next month.

For questions requiring yes/no answers, the way to expand the range of replies is to allow for 'perhaps'. Once this step is taken, then you have to allow for a range for 'perhaps' extending from 'almost certainly no' to 'almost certainly yes'. The easiest way to accommodate this is to provide the user with a scale of numeric inputs. For example, zero could be a definite 'no', five a definite 'yes', and any number in between a 'maybe', with the user selecting his own confidence level as to how close his answer should be to yes or no.

Last month I gave you an expert system to select the most appropriate computer from a list of features. Features were selected either as yes or no requirements, with no room for a 'maybe' answer. That, of course, could be provided if a 'yes' answer' could be considered to mean 'essential', 'no' to be 'not required', and any response in between to be 'desirable', with a numeric value between the maximum and minimum provided to indicate the degree of desirability.

Rather than repeat the computer selection example, this



Figure 1: Expert system for essential and desirable features.

```
100 REM      Expert System
110 REM   to advise on QL Books
120 REM   from essential and
130 REM    desirable features
140 REM
150 REM     by David Nowotnik
160 REM     February, 1987
170 REM
180 REM    Initialise
190 REM
200 RESTORE : LET b=21: LET f=13
210 DIM b$(b,45): DIM a$(b,45): DIM o(b,3)
220 DIM r$(b,f): DIM q$(f,26)
230 FOR i=1 TO b
240 READ b$(i),a$(i),r$(i)
250 NEXT i
260 FOR i=1 TO f
270 READ q$(i)
280 NEXT i
290 REM
300 DATA "Quill,Easel,Archive&Abacus on the Sinclair QL"
310 DATA "Alison McCallum-Varey, £6.95"
320 DATA "0000111101000"
330 DATA "Mathematics on the Sinclair QL"
340 DATA "Czes Kosniowski, £6.95"
350 DATA "0100000000010"
360 DATA "Inside the Sinclair QL"
370 DATA "Jeff Naylor & Diane Rogers, £6.95"
380 DATA "0010000010001"
390 DATA "Word Processing with the Sinclair QL"
400 DATA "Mike O'Reilly, £6.95"
410 DATA "0000100001001"
420 DATA "Professional and Business Uses of the Sinclair QL"
430 DATA "Colin Lewis, £7.95"
440 DATA "0000111101000"
450 DATA "Sinclair QL in Business"
460 DATA "Arnold Handley"
470 DATA "0000111101000"
490 DATA "QL SuperBASIC: A Programmer's Guide"
500 DATA "John Wilson, £6.95"
510 DATA "1100000000110"
520 DATA "Good Programming with QL SuperBASIC"
530 DATA "Roy Atherton"
540 DATA "1100000000000"
550 DATA "QL Easel"
560 DATA "Alison Spottiswoode, 7.95"
570 DATA "0000010001000"
580 DATA "Using Graphics on the Sinclair QL"
590 DATA "Garry Marshall, £7.95"
600 DATA "1100010000000"
610 DATA "Introduction to SuperBASIC on the Sinclair QL"
620 DATA "Dick Meadows, £6.95"
630 DATA "1100000000010"
640 DATA "The QL Book of Games"
650 DATA "R.G. Hurley & David Virgo, £6.95"
660 DATA "1000000000100"
670 DATA "Using the Sinclair QL"
680 DATA "Eric Deeson"
690 DATA "1000111100011"
700 DATA "Maximise Your QL"
710 DATA "Susan Curran & Ray Curnow, £8.95"
720 DATA "1010111111011"
730 DATA "A QL Compendium"
740 DATA "Martin Gandoff & Robin Kinge, £7.95"
750 DATA "1100000001110"
760 DATA "QL Quill"
770 DATA "F Simon & C Spottiswoode, £7.95"
780 DATA "0000100001000"
790 DATA "Developing Applications on the Sinclair QL"
800 DATA "Mike Grace, £6.95"
810 DATA "0000111101000"
820 DATA "The Sinclair QDOS Companion"
830 DATA "Andrew Pennell, £6.95"
840 DATA "0001000010011"
850 DATA "Machine Code Programming on the Sinclair QL"
860 DATA "Martin Gandoff, £7.95"
870 DATA "0011000000000"
880 DATA "Word Processing on the Sinclair QL"
890 DATA "David Dempster, £7.95"
900 DATA "0000100000000"
910 DATA "Advanced Programming with the Sinclair QL"
920 DATA "Martin Gandoff, £6.95"
930 DATA "0100000001010"
1000 REM Enter selections
1010 CLS
1020 PRINT : PRINT "     Select Book Features"
1030 PRINT "  Press 0 (not required) to 5": PRINT "
(essential) for each feature"
1040 DIM q(f)
1050 PRINT
1060 FOR i=1 TO f
1070 PRINT q$(i);" ";
1080 INPUT #0,"Enter 0-5 ";n
1090 IF n<0 OR n>5 OR n <> INT (n) THEN GO TO 1080
1100 PRINT n
1110 LET q(i)=n
1120 NEXT i
1125 PRINT : PRINT " Please wait - I'm thinking!"
1130 REM  Check essentials
1140 LET tot=0
1150 FOR i=1 TO f
1160 IF q(i)=5 THEN  LET tot=tot+1
1170 NEXT i
1180 IF tot=0 THEN  GO SUB 6000: GO TO 1320
1190 FOR i=1 TO b
1200 LET n=0
1210 FOR j=1 TO f
1220 IF r$(i,j)>"0" AND q(j)=5 THEN  LET n=n+1
1230 NEXT j
1240 LET o(i,1)=n
1250 NEXT i
1260 REM Do any books have all essentials?
1270 LET essen=0
1280 FOR i=1 TO b
1290 IF o(i,1)=tot THEN  LET essen=essen+1
1300 NEXT i
1310 IF essen=0 THEN  GO SUB 7000
1320 REM Now find the best
1330 FOR i=1 TO b
1340 IF o(i,1)<tot THEN  GO TO 1400
1350 LET x=0
1360 FOR j=1 TO f
1370 LET x=x+q(j)* VAL r$(i,j)
1380 NEXT j
1390 LET o(i,2)=x
1400 NEXT i
1410 FOR i=1 TO essen
1420 LET high=0
1430 FOR j=1 TO b
1440 IF o(j,2)=0 OR o(j,3)>0 THEN  GO TO 1460
1450 IF o(j,2)>high THEN  LET high=o(j,2): LET item=j
1460 NEXT j
1470 LET o(item,3)=i
1480 NEXT i
1490 REM Now Make Recommendation
1500 CLS
1510 PRINT "I recommend the following book";
1520 IF essen>1 THEN  PRINT "s";
1530 PRINT : PRINT : IF essen>3 THEN  LET essen=3
1540 FOR i=1 TO essen
1550 FOR j=1 TO b
1560 IF o(j,3) <> i THEN  GO TO 1600
1570 LET p$=b$(j): GO SUB 5000
1580 LET p$=a$(j): GO SUB 5000
1590 PRINT
1600 NEXT j
1610 NEXT i
1620 STOP
5000 REM Neat printing routine
5010 LET ii=0: LET le= LEN (p$): LET li=0
5020 LET x$=""
5030 LET ii=ii+1: IF ii>le THEN  GO TO 5060
5040 IF p$(ii)=" " THEN  GO TO 5060
5050 LET x$=x$+p$(ii): GO TO 5030
5060 IF li+ LEN (x$)>31 THEN  PRINT : LET li=0
5070 PRINT x$;: LET li=li+ LEN (x$)
5080 IF li<31 THEN  PRINT " ";: LET li=li+1
5090 IF ii<le THEN  GO TO 5020
5100 PRINT : RETURN
6000 REM No essential features
6010 PRINT : PRINT " No essential features given!"
6020 FOR k=1 TO b
6030 LET o(k,1)=1
6040 NEXT k
6050 LET essen=3: LET tot=1
6060 RETURN
7000 REM  No essentials found
7010 PRINT : PRINT " No perfect match found"
7020 LET tot=1: LET essen=3
7040 RETURN
8000 REM   Items
8010 DATA "Beginner's BASIC"
8020 DATA "Advanced BASIC"
8030 DATA "Beginner's Machine Code"
8040 DATA "Advanced Machine Code"
8050 DATA "Quill"
8060 DATA "Easel"
8070 DATA "Archive"
8080 DATA "Abacus"
8090 DATA "QDOS"
8100 DATA "Business Applications"
8110 DATA "Games"
8120 DATA "Maths/Utilities"
8130 DATA "Hardware"
```

month I have chosen QL books to be the subject of my expert system. My apologies to Spectrum users, but as I have a far wider selection of QL books than those for the Spectrum, I feel I can be more convincing as an expert on QL material. Like all of the systems I have presented so far, the knowledge base can be changed to permit the program to be applied to many other applications. So, if anyone has a vast library of Spectrum books, perhaps you'd care to let us have your own knowledge base on Spectrum books.

The listing for the essential/desirable feature expert system appears in Figure 1. It was written on a Spectrum, so will work on unmodified on that micro. Then it was imported into the QL, and tested. Only two minor changes are necessary for the program to work on the QL. In line 1370, VAL is used to convert a string to its numeric value: VAL not a valid QL keyword. Because of the QL's ability to perform coercion, this line works perfectly if VAL is ommited. In line 5050, the Spectrum's string joining operator '+' should be replaced by '&' on the QL.

By looking through the listing you'll see that most of the program lines are DATA statements; the knowledge base (line 300 to 930) and features items (lines 8000 to the end). Lines 200 to 280 contain a short initialisation routine which loads all the data into a number of arrays.

Knowledge base

The variables 'b' and 'f' are the number of books and features, respectively. This expert system contains details on 21 books. There are a lot more available for the QL, but those in the knowledge base represent a reasonable selection, and the system will provide some pretty good advise on choosing a book. Arrays b$, a$, and q$ contain the book titles, authors (and price), rules and feature items, respectively. Array 'o' provides space for storing the intermediate and final results of selection.

When the program is RUN, after initialisation, the 'Enter selection' routine requires the user to enter values between '0' (not required) and '5' (essential) for each of the 13 features. As this system again uses the parallel process, all features are listed, every time, so that all the information is gathered before any attempt is made in coming to a decision. The responses are stored in numeric array 'q'.

The decision process first examines the responses to see how many features were marked 'essential' (lines 1140 to 1170). If the user fails to mark any feature as essential, then all the books in the knowledge base are scored by multiplying each rule value with each feature value (entered by the user) (line 1320 to 1400), and the total for each book used to obtain a rank order (1410 to 1480).

The program regards the essential list as having top priority. If a book doesn't have all the items marked as essential, then it should not be considered further (lines 1260 to 1310). However, it is possible that no book has all the essential features (line 1310), in which case, the program treats all books as equal, and scores them as if no essential features were entered. What results is the best possible selection of books; that which most closely meets the user's requirements. The program tells the user that an exact match between his essential requirement list and the books' features was not found, so he is aware of the limitation of the recommendation.

In those cases where one or more books containing all essential features are found (and marked), books in this sub-set are scored (lines 1320 to 1400) and their rank order found (line 1410 to 1480), and printed onto the screen. To avoid any confusion, the number of books recommended is always limited to three.

As this decision making process is subjective, it is arguable whether the methods chosen are most appropriate for this problem. In the case of selection of a sub-set of books, it could be asked if this step is really necessary. Then later on in the process, is the scoring method (line 1370) the most appropriate.

Picking essentials

On the matter of picking out only those books which have all essential features, can I use an example from recent experience? Right now I am house hunting, gathering details from a large number of estate agents. While I gave the estate agents my list of essential and desirable features, they seem to pay little regard to that list, leaving me to painstakingly sort through reams of house details. Now, if they'd had my expert system, I'd have much less but relevant house details to sort through, and I'd certainly be more favourably disposed to those agents who seemed to be providing me with details on the right sort of house only. And for my book expert system, the same would apply. The user should have just the right books in the computer's list.

As for the scoring method, why not try varying the equation in line 1370 to see if it has any effect.

In fact, as it stands, the expert system isn't all that good. The problem is not the scoring method, but is a result of the rule base. You'll note that the rules are simply a set of ones and zeros, reflecting whether a book has, or has not got any particular feature. It doesn't take an expert to pick up a book and flick through the pages to determine whether items do or do not appear. An expert would be expected to do more than this: to pass some professional judgement on the quality of the book. This can be done by replacing all number ones in the rule base by a number, one to nine, reflecting the quality of each feature within the book. This I have done for this expert system. The alternative rule base appears in Figure 2. You can either replace the rule lines in the listing in Figure 1 by those in Figure 2, or type in Figure 2 as a separate 'program', then MERGE it into the expert system.

Now, what you will have is a true expert system. The rules were 'extracted' from me, as the expert, and placed into the knowledge base, such that the computer is not only able to pick out books with the required features in a fairly mechanical fashion, but will rank the appropriate books against an expert opinion (if you'll allow me to describe myself as an expert!). Compare the modified and unmodified expert systems: if you have any experience of these books, then I hope you'll agree with my judgement (as provided by the computer). And for those with little knowledge of QL books, my expert system should provide a useful guide to selection.

Figure 2: Alternative rules to improve the QL Books expert system.

```
320 DATA "0000555502000"
350 DATA "0300000000080"
380 DATA "0020000020002"
410 DATA "0000500002002"
440 DATA "0000332201000"
470 DATA "0000434403000"
510 DATA "5300000000220"
540 DATA "7400000000000"
570 DATA "0000090004000"
600 DATA "5300050000000"
630 DATA "4200000000020"
660 DATA "3000000000700"
690 DATA "7000443400011"
720 DATA "4020333322011"
750 DATA "5200000002530"
780 DATA "0000800004000"
810 DATA "0000444304000"
840 DATA "0006000060043"
870 DATA "0053000000000"
900 DATA "0000700000000"
930 DATA "0400000003030"
```

In other situations, you can deal with the uncertainty of yes/no/perhaps responses in much the same way. Using definite 'yes' responses or may not allow you to form a sub-set of answers (you'll have to decide whether this is logical in your application), but using a mathematical formula based on rule values and user response values to provide a score of each possible outcome certainly can be applied in any subjective approach to expert system decision making. If your expert is sufficiently knowledgeable, he may provide a suitable formula.

Figure 3: Demonstration of probabilities.

a) Spectrum Listing

```
100 REM Demonstration of
110 REM Probability Determination
120 REM   by David Nowotnik
130 REM    March, 1987
140 REM
150 PAPER 0: CLS : INK 7
160 PLOT 25,160
170 DRAW 0,-120: DRAW 200,0
180 INK 6: DRAW 0,120
190 INK 7
200 FOR i=1 TO 6
210 PLOT 22,40+i*20: DRAW 3,0
220 PRINT  AT 16-i*2.5,1;i
230 NEXT i
240 INK 6
250 FOR i=1 TO 10
260 PLOT 225,40+i*12
270 DRAW 3,0
280 PRINT  AT 16-i*1.5,29;i/10
290 NEXT i
300 INK 7
310 PRINT  AT 20,0;"Less than 6 (1), Less than 4 (2)Equal to
3 (3). Press 1,2, or 3"
320 LET z$= INKEY$ : IF z$="" THEN  GO TO 320
330 IF z$<"1" OR z$>"3" THEN  GO TO 320
340 LET z= VAL z$
350 GO SUB 1000
360 IF z>1 THEN  GO TO 410
370 LET a$="<6"
380 PRINT  AT 19,4;"Less than six"
390 GO SUB 600
400 GO TO 490
410 IF z <> 2 THEN  GO TO 460
420 LET a$="<4"
430 PRINT  AT 19,4;"Less than four"
440 GO SUB 600
450 GO TO 490
460 LET a$="=3"
470 PRINT  AT 19,4;"Equal to three"
480 GO SUB 600
490 GO SUB 1000
500 PRINT  AT 20,5;"Again? (y/n)"
510 LET z$= INKEY$ : IF z$="" THEN  GO TO 510
520 IF z$="y" THEN  RUN
530 STOP
540 REM
600 REM Plot Graph
610 DIM f(6)
620 LET tot=0: LET v=0
630 FOR i=26 TO 224
640 LET n= INT ( RND *6)+1
650 LET f(n)=f(n)+1
660 PLOT i,40+n*20
670 LET n$= STR$ n: LET n$=n$+a$
680 LET x= VAL n$
690 IF x THEN  LET v=v+1
700 LET tot=tot+1
710 INK 6: PLOT i,40+(v/tot)*120
720 INK 7
730 PRINT  AT 20,0;"Sample = ";i-25;" Random number = ";n
740 PRINT "  Occurance = ";v/tot;"          "
750 PAUSE 20
760 NEXT i
770 FOR i=1 TO 6
780 PRINT  AT 16-i*2.5,4;f(i)
790 NEXT i
800 PAUSE 100
810 RETURN
990 STOP
1000 REM Clear last 3 lines
1010 PRINT  AT 19,0;
1020 FOR k=1 TO 12
1030 PRINT "          ";
1040 NEXT k
1050 RETURN
```

b) QL Listing

```
100 REMark        Demonstration of
110 REMark          Probabilities
120 REMark   David Nowotnik, March 1987
130 :
140 MODE 4
150 WINDOW #1,512,256,0,0: PAPER 4: CLS
160 WINDOW #1,448,200,32,16
170 PAPER 0: INK 7: CLS
180 SCALE 100,-10,-5
190 LINE 140,0 TO 0,0 TO 0,90
200 FOR i=0 TO 90 STEP 15
210   LINE -2,i TO 0,i
220   CURSOR 4,184-i*2: PRINT i/15
230 END FOR i
240 INK 2
250 LINE 140,0 TO 140,90
260 FOR i=0 TO 90 STEP 9
270   LINE 140,i TO 142,i
280   CURSOR 415,184-i*2: PRINT i/90
290 END FOR i
300 WINDOW #0,448,30,32,224
310 PAPER #0,0: INK #0,7: CLS #0: CSIZE #0,1,0
320 PRINT #0\\"  Less than 6 (1), Less than 4 (2), or Equal
to 3 (3)"
330 PRINT #0,," Press 1,2, or 3"
340 REPeat loop1
350   z$=INKEY$(-1)
360   IF z$>"0" AND z$<"4" THEN EXIT loop1
370 END REPeat loop1
380 CLS #0
390 z=z$
400 SELect ON z
410 =1
420 PRINT #0,"    Less than Six"
430 GRAPH 1,2,3,4,5
440 =2
450 PRINT #0,"    Less than Four"
460 GRAPH 3,2,1,3,2
470 =3
480 PRINT #0,"    Equal to Three"
490 GRAPH 3,3,3,3,3
500 END SELect
510 PRINT #0,," Again? (y/n)"
520 REPeat loop
530   z$=INKEY$(-1)
540   IF z$="y" OR z$="n" THEN EXIT loop
550 END REPeat loop
560 IF z$="y" THEN RUN
570 STOP
580 :
590 DEFine PROCedure GRAPH (a,b,c,d,e)
600 DIM f(6)
610 tot=0: v=0
620 FOR i=1 TO 140
630   n= RND(1 TO 6): f(n)=f(n)+1
640   INK 7: POINT i,n*15
650   tot=tot+1
660 IF n=a OR n=b OR n=c OR n=d OR n=e THEN v=v+1
670   INK 2: POINT i,(v/tot)*90
680   AT #0,1,1: PRINT #0,"Sample = ";i;" Random Number =
";n;" Occurance = ";v/tot;"         "
690   PAUSE 20
700 END FOR i
710 INK 7
720 FOR i=1 TO 6
730   CURSOR 32,184-i*30
740   PRINT f(i)
750 END FOR i
760 END DEFine
```

Otherwise, it will be a matter of trying a number of formulae, seeing which gives the best response.

Playing around with mathematics to get the best fit will not be allowed (if you or your user are purists) if you decide to bring in statistics and probabilities into the decision algorithms of your expert system. The basics of probability theory are well established, and standard formulae exist for problem solving with these theories.

Laws of probability

As I promised at the beginning, I'll give you an introduction into the use of probabilities in dealing with uncertainty in an expert system. In a page or so, I can hardly do justice to a subject which can be spread across several volumes, especially as my knowledge of the area extends over just a few chapters! So, what you will get is a brief introduction to some of the theory of probability, then I will give you a relatively short program to demonstrate some of the principles. In the next two parts of this series there will be more substantial examples of the use of probabilities in expert systems.

To begin, let me define probability. In every day speech we use it in an abstract sense, using a phrase like "probably it will". We are saying that, from our personal experience, an event has a greater chance of taking place than not taking place. Take the example of tossing a coin. We know, almost

intuitively, that the chances of getting heads is equal to getting tails: that chance is 50%.

For something we are uncertain about, we can take observations, and use the results of those observations to work out what is likely to happen in the future. The probability is calculated from the number of times an event has taken place compared to the number of observations. Put mathematically, the probability of an event taking place (we call that P(E)) is calculated as:

$$P(E) = \frac{\text{Number of times event occurs}}{\text{Total number of observations}}$$

A P(E) value of 1 would indicate that an event is certain to take place, a P(E) value of zero means that an event is certain **not** to take place. So, as probability P(E) must have values between one and zero, and the larger the P(E) value within this range, the more likely an event is to take place.

Take for example, rolling a dice. Very much like the coin example, you probably already know the chance of rolling a particular value, or range of values — you are an expert in dice rolling! But, for the sake of argument, say you didn't know the odds, and you wanted to find out by observation. The program listings in Figure 3 provide a simulation of that situation.

Figure 3 contains separate listings for Spectrum and QL, to allow for the different graphic displays offered by both computers. Once you have typed in the listing appropriate for your machine, SAVE and RUN it.

At first, you will see a blank piece of graph paper drawn, with two vertical axes. On the left, the axis is marked one to six, each one of the possible outcomes of rolling a dice. The right hand axis is marked as zero to one in units of 0.1, and denotes probability.

Roll of the dice

The program provides three examples, plotting the observation of obtaining a number less than six, less than four, or equal to three. When you make your selection, by pressing 1,2, or 3, up to 200 observations are made of rolling a dice. This is a random event, and makes use of the computer's random number generator. As each number is selected, then a white dot (pixel) is plotted against its value on the left hand axis; this gives you a visual check of the frequency a number is selected at random. Once all of the samples have been 'observed', then numbers are printed against the left hand axis, to indicate the number of times each face of the dice was observed.

But the more important graph is the one which appears in red on the QL, (yellow on the Spectrum). This keeps a running total of the number of times the event (be it less than 6, less than 4, or equal to 3) has taken place, divided by the total number of observations. This, of course, is the formula for calculating P(E), and what this graph is doing is showing you effect on P(E) of taking more and more observations.

A quick calculation might tell you that the probability for throwing a dice value less than six is 5/6 or 0.833. Similarly, for values less than four, the probability is 3/6 or 0.5, and the probability for three being thrown is 1/6, or 0.166. With those facts, use the program to see how well you do by observation.

Generally, with few observations, P(E) is very unreliable, but by the time you have taken around 25-50 observations, the graph should demonstrate something reasonably close to the expected values. Statistics could be used to express confidence (as a numerical value) in the precision of your observation technique, but that's one diversion we won't be taking now!

Allowing for bias

Observing the roll of dice is relatively trivial compared with taking observations of real situations; the problem is bias. For example, say you wanted, for an expert system on car repairs, to know the frequency that your car would start on a damp morning. When is a morning 'damp' and when is it not damp? The result will be biased by your personal assessment of dampness. Does the time of year (and hence temperature) influence the result? And once you have made your observations (over several weeks), then the probability value you obtain will be specific for your car. An expert system for cars in general will need to have observations made on a large number of cars to extract general trends, and determine general probabilities.

All of which goes to show that you have to be very careful in taking observations if the results at the end of the exercise are to be valid. And that is a subject which can (and does) occupy several volumes of books, if you really wanted to get into the theory of sampling. But, for brevity, just be warned to watch out for bias if you do want to try to work out more probabilities for yourself by observation.

Once you have some basic probability values, then there are quite a few mathematical manipulations you (or the computer expert) can do with them.

For a start, the probability of an event **not** taking place is equal to one minus the probability of that event taking place, ie

P(not E)=1 — P(E)

Quite logically, really! If you have two quite independent events, let's call them A and B, then the probability of either one or the other event taking place is the sum of the two probabilities,

P(A or B)=P(A) + P(B)

The probability of **both** these events taking place is the product of the two probabilities:

P(A and B)=P(A) x P(B)

The important rule for these relationships is that the events are fully independent. For example, these formulae would be valid if you wanted to know the combined probability of two throws of a dice. The first throw will have absolutely no influence on the second throw. But if you wanted to know the probabilities of pulling two particular cards from a pack of cards, then the mathematics becomes more complicated. The probability of pulling the Ace of Spades on the second attempt will be zero if it was removed from the pack on the first pull. If that Ace is still in the pack after the first card is removed, then the probability of pulling the Ace of Spades is 1/51.

The way of expressing conditional probability is P(A!B); the probability of event A given that event B has already occured. Under such circumstances, then:

P(A and B)=P(A) x P(B:A)

That is, the probability of both events A and B occuring is the product of the probability for event A times the probability of event B, given that event A has already occured.

Of course, here event A is fully dependent upon event B. There can be situations where, for example, event A is only partly dependent upon event B. That, for now, is introducing unnecessary complication, but it is worth being aware that, under these circumstances, these simple relationships in probability no longer apply.

All that may be fine in the abstract, but what of the real world, and the influence of probabilities on the decision of experts, you may well ask. Well, next month I'll be taking probabilities one step further into expert sytems, and taking a look at some more strategies for reaching decisions. And I will examine how to select questions more rationally.

Comic Strip



RANDOM MEMORY

Clyde Bish ventures farther into his Comic Strip utility.

This month we add an editor program to unravel the number strings and get that necessary data into memory with the minimum of fuss. (If you've managed OK in the intervening weeks then (a) You don't need my editor program and (b) YOU should be writing for ZXC!)

But before you set it to work, a timely warning. Don't be in too much of a rush to get stuck in. The editor will only help you to load the strings into memory. It **won't** make them for you. That you must do.

You must start with a good idea of what your strip is to be about. Don't aim too high for starters. A short sequence that, perhaps you can add to later would be ideal to get you going. (I experimented with the opening sequences from a "Fighting Fantasy" book). Next design the outline drawings, icons, captions etc. on pixel paper. Again, don't make your pictures too complex. Remember you're only working in an 80x64 pixel box so the final resolution is quite low.

Some adventure illustrators go for highly detailed pictures which might look good on paper, but transform into a mess on screen. (If you have a Monitor, or a non-"pixel creep" Spectrum you can get away with far more). Rather go for bold line. After all, it's atmosphere you're trying to create. Not Art Nouveau!

Now convert each drawing into a series of PLOT/DRAW co-ordinates, the embellishments into their respective PRINT AT co-ordinates, and characters, and produce UDG banks on tape for the icons — perhaps using a designer program. (Remember icon type 0 uses UDGs "a" through "l" in a 3x4 matrix, whilst type 1 "a" to "t" in a 4x5 block, and type uses *two* "a" to "r" sets, each in a 6x3 pattern for top and bottom).

You will now know how long each data block needs to be — an important point when setting each data start address later.

Assuming you've done this preparatory work **Program 1** will help you out, so type it in. The groups of capital letters are UDGs and should be entered in a mode. Hopefully as you do so you'll notice that I have included few error checks. This is partly to make the printed listing shorter (and so leave space for other contributors!) but also because it's a good exercise for you to write your own in. **Program 2** is an example of how to trap an input other than a number between 176 and 255 — the x co-ordinate range for the drawCODE. If you need more help then refer to Alan Davis' article on Friendly Programming in the February issue of ZXC. (I personally prefer to leave the CAPS SHIFT 6 "Stop in INPUT" error available in case I WANT to escape! Anyway, any smart alec who uses that trick deserves to have his program crash!). If you decide not to put in traps and you get an error message, note the line number that accompanies it and examine the listing there. You can usually work out a restart GOTO which will leave your code intact. The "Panic Button" is GOTO 100, which will return you to the menu.

Now back to the program. RUN 9000 to start it up, and you'll be asked for the address for the start of your data. Enter this and you'll get the menu on screen, plus the present start address. This will appear at the start of each of the Create Data options and can be altered. You may decide it's easier to put the data blocks in backwards i.e. using the highest memory block first and moving back through the memory the required number of bytes each time, rather than working from low to high with the whole data block. If so you can use menu option 7, which, given the length of the code block to be entered, will tell you where to set the start address. Don't try to mix up up both systems or you'll get into an awful muddle!

Now to the options:—

1. Create display drawcode

This, you remember, draws the outline picture. **Table A** gives a suitable data block to test. (It produces the "end of corridor" outline from last month's screen dump). You begin by deciding on the start address as outlined above. If you press just ENTER it will remain as set. The new start address will be displayed in decimal and in low/high byte

Table A

176	175	199	168	232	168	232	167
199	127	199	168	1	233	168	255
175	1	232	127	255	112	1	199

RANDOM MEMORY

Comic Strip



form. This latter form you will need to POKE into the drawCODE machine code before calling the routine so make a note of them. You will also get an input cue for the co-ordinate of the first PLOT position. Pressing just ENTER now will allow you to escape to the menu when you've finished. (It also allows you to escape if you've got here by accident! This facility is available on all options).

When you input a number the cue changes to "y co-ordinate?". After that input the cue becomes "DRAW x co-ordinate?", followed by "y co-ordinate?". Follow these as you enter the sequence of DRAW co-ordinates from the first PLOT position.

When you want a new PLOT position, enter a 1 for the first DRAW co-ordinate and you'll get back to PLOT. To end the string enter a 0 for the DRAW x co-ordinate and you'll be given the option to display your drawcode. Pressing Y will give a box top right, the size of your cartoon drawing. Subsequently

PROGRAM 1

```
 100 POKE 23693,56: CLS : PRINT
"Present Start Address: ";a''"OP
TIONS"''"1 CREATE/DISPLAY DRAWCO
DE"'"2 CREATE/DISPLAY STRINGCODE
"'"3 CREATE/DISPLAY ICONCODE"'"4
 CREATE CONTROLCODE"'"5 CALCULAT
E CIRCLE  DATA "'"6 SAVE CODES"'
"7 CALCULATE ADDRESS"'''"Press n
umber"
 110 PAUSE 0: LET r$=INKEY$: CLS
 : GO TO VAL r$*1000
1000 INPUT "Start address? (set
at ";(a);")"' LINE a$: IF a$<>""
 THEN  LET a=VAL a$
1004 LET a1=a
1005 CLS : RANDOMIZE a: PRINT "N
ew start address:"'a'"(";PEEK 23
670;"/";PEEK 23671;")"'': LET a2
=a
1010 INPUT "Plot x?", LINE a$: I
F a$="" THEN  LET d=1: GO SUB 61
00: GO TO 100
1020 POKE a,VAL a$: PRINT a,a$:
LET a=a+1: INPUT "PLOT y?",y: PO
KE a,y: PRINT a,y: LET a=a+1
1030 INPUT "DRAW x?",x: POKE a,x
: PRINT a,x: LET a=a+1: IF x=1 T
HEN  GO TO 1010
1035 IF x=0 THEN  GO TO 1500
1040 INPUT "DRAW y?",y: POKE a,y
: PRINT a,y: LET a=a+1: GO TO 10
30
1500 PRINT '"Display Drawcode?":
 PAUSE 0: IF PEEK 23556<>89 THEN
  GO TO 1005
1501 GO SUB 1504: GO TO 1005
1504 CLS
1505 PLOT 176,175: DRAW 79,0: DR
AW 0,-63: DRAW -79,0: FOR f=0 TO
 7: PRINT AT f,21;" ": NEXT f
1510 PLOT PEEK a2,PEEK (a2+1): P
RINT #0;AT 0,0;"PLOT"'a2;"   x="
;PEEK a2,"y=";PEEK (a2+1): PAUSE
 0: IF PEEK 23556=13 THEN  GO TO
 100
1520 INPUT ;: LET a2=a2+2
1530 DRAW PEEK a2-PEEK 23677,PEE
K (a2+1)-PEEK 23678: PRINT #0;AT
 0,0;"DRAW"'a2;"   x=";PEEK a2,"
y=";PEEK (a2+1): PAUSE 0: IF PEE
K 23556=13 THEN  GO TO 100
1540 INPUT ;: LET a2=a2+2: IF PE
EK a2>1 THEN  GO TO 1530
1550 IF PEEK a2<>0 THEN  LET a2=
a2+1: GO TO 1510
1560 PRINT #0;AT 0,0;"END OF DRA
WCODE": PAUSE 250: RETURN
2010 INPUT "Start address? (set
at ";(a);")"' LINE a$: IF a$<>""
 THEN  LET a=VAL a$
2011 LET a1=a
2020 CLS : RANDOMIZE a: PRINT AT
 0,0;"Start Address:"'a'"(";PEEK
 23670;"/";PEEK 23671;")": INPUT
 "Address of DRAWcode?", LINE r$
: IF r$<>"" THEN  LET a2=VAL r$:
 GO SUB 1505
2023 IF r$="" THEN  GO TO 100
2025 PLOT 176,175: DRAW 79,0: DR
AW 0,-63: DRAW -79,0: FOR f=0 TO
 7: PRINT AT f,21;" ": NEXT f
2040 INPUT "Row?",r: INPUT "Colu
mn?",c
2050 INPUT "Text?",r$: PRINT AT
r,c;r$
2055 POKE a,22: POKE a+1,r: POKE
 a+2,c: LET a=a+3: FOR f=1 TO LE
N r$: POKE a,CODE r$(f): LET a=a
+1: NEXT f
2060 PRINT #0;AT 0,0;"More?": PA
USE 0: IF PEEK 23556=78 THEN  LE
```

```
T d=1: GO SUB 6100: GO TO 100
2070 PRINT #0;AT 0,0;"Same strin
g?": PAUSE 0: IF PEEK 23556=78 T
HEN  GO TO 2020
2080 GO TO 2030
3010 INPUT "Start address? (set
at ";(a);")"' LINE a$: IF a$<>""
 THEN  LET a=VAL a$
3020 LET a1=a: INPUT "Icon type?
", LINE a$: IF a$="" THEN  GO TO
 100
3025 GO TO 3050+VAL a$*10
3050 CLS : RANDOMIZE a:: INPUT "
Title for udg set?"' LINE r$: LO
AD r$CODE USR "a": PRINT "Addres
s:"'a'"(";PEEK 23670;"/";PEEK 23
671;")"''"Type 0: 3x4"''"      "
'"     "'"     "'"     "''"O.
K.?": PAUSE 0: IF PEEK 23556=78
THEN CLS : GO TO 3050
3055 PRINT '"Transferring bytes"
: FOR n=0 TO 7: FOR f=USR "a"+n
TO USR "l"+n STEP 8: POKE a,PEEK
 f: LET a=a+1: NEXT f: NEXT n
3059 GO TO 3100
3060 CLS : RANDOMIZE a: INPUT "T
itle for udg set?"' LINE r$: LOA
D r$CODE USR "a": PRINT "Address
:"'a'"(";PEEK 23670;"/";PEEK 236
71;")"''"Type 1: 4x5"''"
"'"      "'"      "'"

 "'"      "''"O.K.?": PAUSE 0:
 IF PEEK 23556=78 THEN  CLS : GO
 TO 3060
3065 PRINT '"Transferring bytes"
: FOR n=0 TO 7: FOR f=USR "a"+n
TO USR "t"+n STEP 8: POKE a,PEEK
 f: LET a=a+1: NEXT f: NEXT n
3069 GO TO 3100
3070 CLS : RANDOMIZE a: INPUT "T
itle for udg set 1?"' LINE r$: L
OAD r$CODE USR "a": PRINT "Addre
ss:"'a'"(";PEEK 23670;"/";PEEK 2
3671;")"''"Type 2: 6x6"''"
   "'"       "'"

  "''"O.K.?": PAUSE 0: IF PEEK
 23556=78 THEN  CLS : GO TO 3070
3075 PRINT ''"Transferring bytes
": FOR n=0 TO 7: FOR f=USR "a"+n
 TO USR "r"+n STEP 8: POKE a,PEE
K f: LET a=a+1: NEXT f: NEXT n
3080 INPUT "Title for udg set 2?
"' LINE r$: LOAD r$CODE USR "a":
 PRINT AT 9,0;"          "'"
     "'"      "''"O.K
.?                      ": PAUSE 0
: IF PEEK 23556=78 THEN  GO TO 3
080
3085 PRINT '"Transferring bytes"
: FOR n=0 TO 7: FOR f=USR "a"+n
TO USR "r"+n STEP 8: POKE a,PEEK
 f: LET a=a+1: NEXT f: NEXT n
3100 LET d=255: GO SUB 6100: PRI
NT '"More?": PAUSE 0: IF PEEK 23
556=78 THEN  GO TO 100
3110 CLS : GO TO 3020
4010 INPUT "Start address? (set
at ";(a);")"' LINE a$: IF a$<>""
 THEN  LET a=VAL a$
4015 LET a1=a
4020 CLS : RANDOMIZE a: PRINT "S
tart address:"'a'"(";PEEK 23670;
"/";PEEK 23671;")"'': INPUT "No.
 of Strings", LINE r$: IF r$=""
THEN  GO TO 100
4030 LET n=VAL r$: PRINT a,n: PO
KE a,n: LET a=a+1: IF n=0 THEN
GO TO 4100
4040 FOR f=1 TO n: INPUT "Length
 of String ";(f);"?"'l: POKE a,l
: PRINT a,l: LET a=a+1
4050 INPUT "Start address for St
ring ";(f);"?"'s: GO SUB 4999: N
EXT f
4100 INPUT "No. of Icons",n
4130 PRINT a,n: POKE a,n: LET a=
a+1: IF n=0 THEN  GO TO 4200
4140 FOR f=1 TO n: INPUT "Row fo
r Icon ";(f);"?"'l: POKE a,l: PR
INT a,l: LET a=a+1
4150 INPUT "Column for Icon ";(f
);"?"'l: POKE a,l: PRINT a,l: LE
T a=a+1
4160 INPUT "Start Address for Ic
on ";(f);" data?"'s: GO SUB 4999
4170 INPUT "Icon ";(f);" type?"'
l: POKE a,l: PRINT a,l: LET a=a+
1: NEXT f
4200 INPUT "Bubble pointer colum
n?"'n: POKE a,n: PRIT a,n: LET a
=a+1: LET d=255: GO SUB 6100: GO
 TO 4020
4999 RANDOMIZE s: POKE a,PEEK 23
670: POKE (a+1),PEEK 23671: PRIN
T a,PEEK a'a+1,PEEK (a+1): LET a
=a+2: RETURN
5000 INPUT "CIRCLE"'"X coordinat
e of centre?"'"(ENTER to exit) "
; LINE x$: IF x$="" THEN  GO TO
0100
5010 LET x=VAL x$: INPUT "Y coor
dinate of centre? ";y
5020 PRINT AT 19,0;x;",";y
5030 INPUT "radius? ",r
5040 PRINT AT 20,0;r
5050 INPUT "spacing? ";m
5060 PRINT AT 21,0;m
5070 LET th=m*PI/r
5080 LET x1=r: LET y1=0
5090 PLOT x+x1,y+y1
5100 LET c=0
5110 FOR i=1 TO r STEP m/2
5120 LET c=c+1: PRINT AT 0,0;c
5130 LET x2=x1*COS th-y1*SIN th
5140 LET y2=x1*SIN th+y1*COS th
5150 DRAW INT (x2-x1+.5),INT (y2
-y1+.5)
5160 LET x1=x2: LET y1=y2
```

▶

```
5170 NEXT i
5180 CIRCLE x,y,r-3
5190 PRINT #0;"Data  Again  Menu
": PAUSE 0: CLS : IF PEEK 23556=
68 THEN  GO TO 5240
5200 IF PEEK 23556=77 THEN  GO T
O 100
5210 PRINT #0;"new coordinates?
(Y/N)": PAUSE 0: IF PEEK 23556=8
9 THEN  GO TO 7000
5220 PRINT AT 19,0;x;",";y: PRIN
T #0;"new radius? (Y/N)": PAUSE
0: IF PEEK 23556=89 THEN  GO TO
5030
5230 GO TO 5040
5240 LET p=2: PRINT "COPY? (Y/N)
": PAUSE 0: IF PEEK 23556=89 THE
N  LET p=3
5250 PRINT #p''"centre coords=";
x;",";y;"  radius=";r''
5260 PRINT #p;"start x,y=";INT (
x+x1+.5);",";INT (y+y1+.5)''
5270 LET c=0
5280 FOR i=1 TO r STEP m/2
5290 LET x2=x1*COS th-y1*SIN th
5300 LET y2=x1*SIN th+y1*COS th
5310 LET x1=x2: LET y1=y2
5320 PRINT #p;TAB c;INT (x+x1+.5
);",";INT (y+y1+.5);
5330 LET c=c+10: IF c>23 THEN  P
RINT '': LET c=0
5340 NEXT i: IF p3 THEN  LPRINT
5350 INPUT "Press ENTER to conti
nue  "; LINE t$: GO TO 5190
6000 PRINT "SAVE"''"Start Addres
s: ";a1,"End address:   ";a-1''"
Length: ";a-a1
6010 INPUT "Title for Code?"' LI
NE t$: IF t$="" THEN  GO TO 100
6020 INPUT "Save from address?",
s
6030 INPUT "Length of Code?",l
6035 INPUT "Erase existing Code?
 "; LINE a$: IF a$<>"" THEN  ER
ASE "m";1;t$: REM for microdrive
 only
6040 PRINT ''"SAVEing:"''t$;" CO
DE ";s;",";l: SAVE t$CODE s,l: P
RINT '"VERIFY": VERIFY t$CODE
6045 PRINT #0;AT 0,0;"Copy?": PA
USE 0: IF PEEK 23556=89 THEN  LP
RINT 't$;" CODE ";s;",";l
6050 GO TO 100
6100 PRINT #0;"Copy to Printer?
(Y/N)": PAUSE 0: IF PEEK 23556<>
89 THEN  RETURN
6110 INPUT "Title? "; LINE t$: L
ET c=0: LPRINT 't$''"Start: ";a1
,"End: ";a-1': FOR n=a1 TO a-1:
IF PEEK n=22 AND d=1 THEN  LPRIN
T '"(";n;")": LET c=0
6115 LPRINT TAB c;PEEK n;: LET c
=c+4: IF PEEK n=0 AND d=1 THEN
LPRINT '"(";n+1;")": LET c=0
6120 IF c=32 THEN  LET c=0
6130 NEXT n: LPRINT : LET d=0: R
ETURN
7000 INPUT "Length of code block
?"'l: PRINT #0;AT 1,0;"Start add
ress: ";a9-l'"(Press ENTER for M
enu)": LET a9=a9-l: PAUSE 0: GO
TO 100
9000 LET d=0: INPUT "Start addre
ss?",a: LET a9=a: GO TO 100
```

pressing any letter will build up your outline drawing whilst displaying the co-ordinates at the bottom of the screen. An END OF DRAWCODE message is given, followed by a pause before you return to the new start address display and "PLOT x co-ordinate?" cue. If you have no more code to set up press just ENTER. You will be given an option to list to printer. If you do this you'll note that each 0 is followed by an address in brackets. This is the start of the new drawcode following. You'll then return to the menu.

2. Create/display stringcode

This holds the information for the embellishments to the line drawings. The option begins asking for start addresses, then displays it to the left of a box outlining the picture area, with the option to display the drawing on which the embellishments will fit. This operates as in option 1. Call up the "dead end" picture you've put in already. (You do remember the data start address, don't you?). You'll then be cued for the row, column and text of the embellishment. Try row 1, column 25, character "@". This will be displayed in the box. Answer the "More?" cue with N and you'll get the option to print out the data (with each 22 — the AT token code — preceded by its address). Otherwise you will be asked if it is the same string, in which case you will add to your picture. Try row 1, column 28, character "@" to put a mystic sign in both corners of the room. Alternatively a new string (with its start address and picture) can be created.

3. Create/display icon

The icons are the illustrations that go onto the background. After the usual address request you'll be asked for the icon type (0, 1 or 2) and the title of the UDG set which holds the data. (You will have made these up beforehand and have them on tape). After the UDGs are loaded the icon will be displayed and, if the response to the "OK?" cue is positive then the bytes will be transferred, beginning at the chosen memory address. You will again have the option of a data printout, and the

PROGRAM 2

```
6010 INPUT "Entry?", LINE i$: IF
 i$="" THEN  GO TO 6000
6015 FOR f=1 TO LEN i$: IF i$(f)
<"0" OR i$(f)>"9" THEN  GO TO 60
10
6020 NEXT f: LET i=VAL i$: IF i<
176 OR i>255 THEN  GO TO 6010
6030 REM :i must hold a number
        between 176 & 255
```

opportunity to continue or escape to the menu. Remember that icon type 2 needs two, 18-character sets of UDGs and so you'll need to load twice.

4. Create control string

This is the string which calls up each element of the picture except the line drawing. It follows the same format as described last month. Again, the usual address request precedes the routine proper.

First you'll be asked for the number of embellishment strings. Replying with a 0 jumps you past this call option. Any other number sets up that loop which asks for the length of a string, and then the address of the start of its data. Enter this as a five figure decimal number and the routine will convert it to its low/high byte form for you.

This call loop is followed by a cue for the number of icons involved. Again 0 bypasses this call whilst any other number sets up an input loop. For each icon you'll be cued for the row and column of the top left corner, the data start address and the icon type.

Finally you'll be cued for the column of the speech bubble pointer. 0 means no speech bubble required.

You will then return to the "strings?" prompt ready to start again. (Note the new address) or press just Enter for the menu.

5. Circle data

This is a very useful option which will calculate the PLOT and DRAW co-ordinates required to draw a circle.

You will be asked for the x and y co-ordinates of the centre of the circle, and the radius in pixels. The "steps?" request relates to the number of steps required to draw the circle. Answering 2 gives the best circle (but requires the most data in memory), whereas a larger number may give a polygon (which, for a large radius, may still be acceptable). A true circle is draw just inside for comparrison. If you want to try again press A. D gives you a list of the required co-ordinates to screen or printer. M returns you to the menu.

6. Save

The start and end addresses, and the length of the current data block is displayed. Enter the title, start address and length of the data block (which can be different from that displayed if you want to save other data) and the routine will save and verify the data code before returning you to the menu.

So much for the pictures, but, as yet, we have no captions. Now is the time to remedy that but first, a little more thought. Each picture is only 10 characters wide so with three

Table B

0	0	0	0	0	0	0	0
0	32	32	32	32	0	32	0
0	40	80	0	0	0	0	0
0	80	248	80	80	248	80	0
32	120	160	112	40	168	112	32
0	200	208	32	64	152	152	0
0	32	80	32	88	144	104	0
0	16	32	0	0	0	0	0
0	16	32	32	32	32	16	0
0	64	32	32	32	32	64	0
0	80	32	248	32	80	0	0
0	32	32	248	32	32	0	0
0	0	0	0	0	32	64	0
0	0	0	248	0	0	0	0
0	0	0	0	0	0	64	0
0	8	8	16	32	64	128	128
0	112	136	152	168	200	112	0
0	32	96	32	32	32	112	0
0	112	136	8	112	128	248	0
0	112	136	48	8	136	112	0
0	16	48	80	144	248	16	0
0	248	128	240	8	136	112	0
0	112	128	240	136	136	112	0
0	248	8	16	32	64	64	0
0	112	136	112	136	136	112	0
0	112	136	136	120	8	112	0
0	0	32	0	0	32	0	0
0	0	32	0	0	32	64	0
0	32	64	128	64	32	0	0
0	0	248	0	248	0	0	0
0	32	16	8	16	32	0	0
0	112	136	16	32	0	32	0
0	112	152	168	184	128	120	0
0	32	80	136	248	136	136	0
0	240	136	240	136	136	240	0
0	112	136	128	128	136	112	0
0	224	144	136	136	144	224	0
0	248	128	240	128	128	248	0
0	248	128	240	128	128	128	0
0	112	136	128	184	136	112	0
0	136	136	248	136	136	136	0
0	112	32	32	32	32	112	0
0	8	8	8	8	136	112	0
0	144	160	192	160	144	136	0
0	128	128	128	128	128	248	0
0	136	216	168	136	136	136	0
0	136	200	168	168	152	136	0
0	112	136	136	136	136	112	0
0	240	136	136	240	128	128	0
0	112	136	136	168	152	112	0
0	240	136	136	240	144	136	0
0	112	128	112	8	136	112	0
0	248	32	32	32	32	32	0
0	136	136	136	136	136	112	0
0	136	136	136	136	80	32	0
0	136	136	168	168	216	136	0
0	136	80	32	32	80	136	0
0	136	136	80	32	32	32	0
0	248	16	32	64	128	248	0
0	48	32	32	32	32	48	0
0	128	128	64	32	16	8	0
0	96	32	32	32	32	96	0
0	32	112	168	32	32	32	0
0	0	0	0	0	0	0	252
0	96	144	128	224	128	240	0
0	0	112	8	120	136	120	0
0	128	128	240	136	136	240	0
0	0	112	136	128	136	112	0
0	8	8	120	136	136	120	0
0	0	112	136	240	128	120	0
0	48	64	96	64	64	64	0
0	0	120	136	136	120	8	112
0	128	128	240	136	136	136	0
0	32	0	96	32	32	112	0
0	16	0	16	16	16	16	96
0	128	128	160	192	160	144	0
0	64	64	64	64	64	48	0
0	0	80	168	168	168	168	0
0	0	240	136	136	136	136	0
0	0	112	136	136	136	112	0
0	0	240	136	136	240	128	128
0	0	112	136	136	120	8	12
0	0	48	72	64	64	64	0
0	0	112	128	112	8	240	0
0	32	112	32	32	40	16	0
0	0	136	136	136	136	112	0
0	0	136	136	136	80	32	0
0	0	168	168	168	168	80	0
0	0	136	80	32	80	136	0
0	0	136	136	136	120	8	112
0	0	248	16	32	64	248	0
0	48	32	64	32	32	48	0
0	8	8	8	8	8	8	0
0	96	32	16	32	32	96	0
0	0	0	64	168	16	0	0
120	132	160	164	164	160	132	120
62	90	42	75	92	190	40	8
245	205	184	25	235	241	24	245
35	78	35	70	35	11	11	197
70	35	78	35	121	214	40	48
252	198	40	135	79	135	129	79
120	254	22	56	2	82	21	135
135	135	214	167	237	68	71	237
57	128	92	209	122	179	200	27
213	126	35	229	111	38	0	41
41	41	17	19	249	28	8	8
14	6	86	203	34	56	38	0
13	32	248	35	16	242	237	78
125	92	62	6	129	79	254	252
58	12	14	0	198	214	8	81
254	247	32	0	5	58	237	67
125	92	225	24	190	245	197	213
229	42	125	92	82	8	145	133
79	160	132	71	205	233	34	225
209	193	241	24	195			

PROGRAM 3

```
10 FOR f=64019 TO 64935: INPUT
i: PRINT f,i: INPUT f,i: NEXT f
```

PROGRAM 4

```
  10 LET z=0: LET p=1: LET m=1:
LET m$="": PRINT #0;AT 0,0;"Hard
 Copies?": PAUSE 0: IF PEEK 2355
6=89 THEN  LET z=1
  20 PRINT "Pointer = ";p;" for
Message ";m: IF z=1 THEN  LPRINT
 '"Message ";m;" Pointer = ";p''
  25 INPUT "No. lines?", LINE l$
: IF l$="" THEN  GO TO 50
  28 LET m$=m$+l$: LET p=p+1: FO
R f=0 TO VAL l$-1: PRINT AT 21,1
1;"V": PRINT AT f+17,0;"Message
";(m);", Line ";(f+1): INPUT  LI
NE a$: IF z=1 THEN  LPRINT a$
  40 LET a$=a$+"          "( T
O 12-LEN a$): PRINT AT f+5,0; BR
IGHT 1;a$: LET m$=m$+a$: LET p=p
+12: NEXT f: PAUSE 50: LET m=m+1
: CLS : GO TO 20
  50 PRINT "Length of m$ = ";p-1
: IF z=1 THEN  LPRINT "Length of
 m$ = ";p-1: STOP
```

Comic Strip



Table C

Message 1. Suddenly a nasty troll appears.
Message 2. You wait . . .
Message 3. You come to a heavy wooden door.
Message 4. You open the green door.

lines of printing below you could only have a maximum of thirty characters which is not very many. It is possible to get more by using a smaller character set. For example "Redhawk" uses half-size characters but that may be a bit on the small size. Let's be truly British and compromise on a 42 character/line set which gives us a few extra spaces per line so longer words are easier to fit in.

Now the bad news. A new character set means data for the new-size characters — 768 bytes of it, plus the code to run it! So take a deep breath and enter the data from **Table B, using Program 3.** Save and verify with

SAVE "P42" CODE 64019, 917

If you also load in "drawCODE" saved from last time you can save both together with:

SAVE "allcode" CODE 64019, 1357

To use it you have to set the string variable z£ to the text, and precede it with the character code for the row and column numbers before calling the machine code. Test it out with:

LET z$=CHR$ 5 + CHR$ 20 + "test": PRINT: RANDOMIZE USR 64787

and you should get the word "test" appear at row 5, column 20 (of 42) i.e. about halfway across the screen.

So now we have the printing technique but still need to hold the data strings for the various captions in the least possible memory space. The technique I've chosen is to hold all the captions in one long string, m$, and use a pointer p to indicate the start byte of each message. This will be the number of lines of print in the message (1 to 3) and is followed by that number of blocks of 12 characters — the text of the message. To make the formation of such a string easier, help is again at hand in the form of **Program 4,** a text editor which will help you do the job. As with the picture editor you need to know what your messages are before you begin, so take time out to write some suitable ones, remembering that each must be a maximum of 3 lines, each of only 12 characters. **Table C** gives some possible examples.

When you run **Program 4** you'll get the option to have hard copy. You'll then see displayed the message number you're at and the position of the pointer along the string. You first have to enter the number of lines for that message. Let's try it out with 3. Now you'll get the prompt "Message 1, line 1" with a "V" beneath. Type in the first line of a caption such as "Suddenly a nasty troll appears!" but don't go past the V. Enter this and continue with the second line, and finally the last — just the word "appears!" Your message will appear highlighted on screen before the message/pointer display reappears, reset to the new value. Note the pointer value if you have no hard copy. (Incidentally, it's worth saving a one-line message of just 12 blank spaces as there will be occasions when you want a picture with no caption). When you have no further captions to code press Enter at the "number of lines?" cue and you'll be told the length of m$. Now save just the variables area (containing m$) by deleting each line then saving with:

SAVE "m$data"

To use this string in your final program you'll be using a routine similar to **Program 5.** Type it in and try it out with some of the pointer values you noted for m$ (and making sure you have P42 code on board and have merged in m$!).

Well, I reckon that's enough mind boggling for one month. Next time we'll get on to moving things around, plus the little refinements such as the clock, scores, persons, icons, etc. Oh yes — AND how it's all going to work! See you then.

PROGRAM 5

```
  5 REM p=start pointer
        m$=text (merged in)
        P42CODE must be on
        board
 10 INPUT "Message pointer?",p
 20 FOR f=1 TO VAL m$(p): LET z
$=CHR$ (f+7)+CHR$ 29+m$(p+1 TO p
+12): PRINT : RANDOMIZE USR 6478
7: LET p=p+12: NEXT f
```



Gremlin Graphics return with an old favourite.

Auf Wiedersehen Monty
Gremlin
£7.95



Auf Wiedersehen MONTY

The software industry seems to be getting a bit nostalgic at the moment. First, Imagine and Gremlin came out with updated versions of Breakout (and CRL have got one on the way too). And now Gremlin have brought about the return of the platform game and Monty Mole. What's more, these old games still seem to have a bit of life left in them, and they're giving a lot of the more 'advanced' software titles a run for their money.

Having survived his last couple of adventures, Monty has decided to go off on a bit of 'round the world sightseeing'. The eighty screens of Auf Wiedersehen Monty are named after various tourist traps, and the screens are arranged in roughly geographical order to give you a rough idea of how to find your way around. The titles of a lot of the screens are irrelevant really. Though some of them do bear some sort of resemblance to the places they're named after (the Pyrenees screen, for instance, is a bit hilly and the monster-sprites are shaped like bulls), most of the screens are just an excuse for a lot of platforms and ladders, crushers, sprites, and deadly spikey things that have to be avoided at all costs. And it's all good fun, as you might expect (not terribly original, I have to admit, but fun all the same).

Of course, a platform game's not a platform game without a few dozen objects to collect, tucked away in various awkward corners. In line with the game's globe-trotting theme the main thing that Monty has to collect is a number of Eurocheques which bump up his money score, but there are also a lot of other objects which have different properties, and it's up to you to find out what does what.

This is where the names of the screens can come in useful. For instance, there's a wheel which might not seem like much use on its own, but when you take it to Monaco (for the Grand Prix) you'll gain extra points. And there are also objects which have some effect on Monty himself — extra lives, alcohol which gets him drunk, even a cheat mode if you find the right object.

It doesn't require state of the art programming skills to write a game like this, but it does take a lot of thought to get the design of the screens right. Many a platform game has died a death because it was just too hard to be enjoyable, but Gremlin have gotten the balance just about right here. The game is hard enough to be addictive and a little bit infuriating, but if you can't get at all the object on the screen it's still reasonably easy for you to move on and explore another part of the game so that you don't get bored by dying in the same place over and over.

The graphics are good — all the sprites are quite large and smoothly animated, and Monty responds well to both keyboard and joystick control.

I've always enjoyed platform games and Gremlin's Monty series has provided some of the better examples of the genre. Apart from a few pretty feeble budget titles there hasn't been a decent platform game released for a while, so Monty's return is very welcome. Now, what with the return of Breakout and Monty Mole, the only question left is — "when does The Return Of Jet Set Willy arrive"?

MONSTER HIT

PROJECT X

Ray Elder unveils the third programming challenge, a professional input routine.

Have you visited your local library recently? If not have you used or peeked over the shoulder of someone using a "professional" computer?

I first noticed the way in which business machines expect input by watching the screen of the system installed in our local library. Most interesting, the information or data is typed and displayed on the screen in various positions after headings which describe what the contents should be.

I had also decided to write a simple program to catalogue my program collection, yes, I'm talking about a database or file program. First let me clarify the terms I'll be using:

FILE — The complete set of all the data on the whole collection.

RECORD — The data which is all the information relating to a single program in my collection.

FIELD — The specific data referring to one part or entry on a record.

For example, the file "Programs" will contain a record for each of my cassettes and on each of these records will be fields for Name, Company, Type, Comment, Price and rating.

Now don't panic! This month's project isn't to write a complete file program, but the problem is to produce *...a professional input routine.*

Not only can this general purpose routine be used for my specific file program but could also be used in any database program or at any point in a game where text and/or number input is required; such as high score name input, adventure game commands, or indeed anything that requires more than a single key press.

1. Identify Problem.

We require the ability to print headings and fields for data entry at any desired screen position, display a cursor and accept and store data entered by the operator. The data must be identified and allocated to the correct field.

The operator should have control of the cursor to move through fields and previously entered data making amendments if required.

2. Breakdown of Process.

This is an exercise which involves several related programming techniques, screen and cursor printing, data input, error trapping, validation and data storage.

The problems can be identified as:

a). Print cursor at y, x position.
b). Move cursor Up and Down from field to field and Left and Right within the current field.
c). Allow the operator to delete or overwrite previous entries.
d). Provide the option to leave a field blank or terminate the entry of data before a field is full. (Use ENTER key?)
e). Provide a means of exiting from the record entry routine.
f). Identify the field being written to and accept only valid characters to the field.

3. Possible Approaches.

This time I have taken the liberty of including a short routine of my own, it is there merely to set up an input screen, I have not *presumed to* include any code for the project, but if you enter it and run it you will get a good idea of what I have been trying to explain (a picture worth a thousand words etc...) and I have also included a screen print — although it is in inverse to the computer display.

The data structure takes the form of Field title, y position, x position, fieldlength and field type, the latter being "N" for numerical and "C" for alphanumeric character, and is useful when writing the validation part to determine whether a field can accept letters and numbers or number only.

These details, especially the field length and type, may be read into an array for future reference or the data simply RESTOREd and re-read as required.

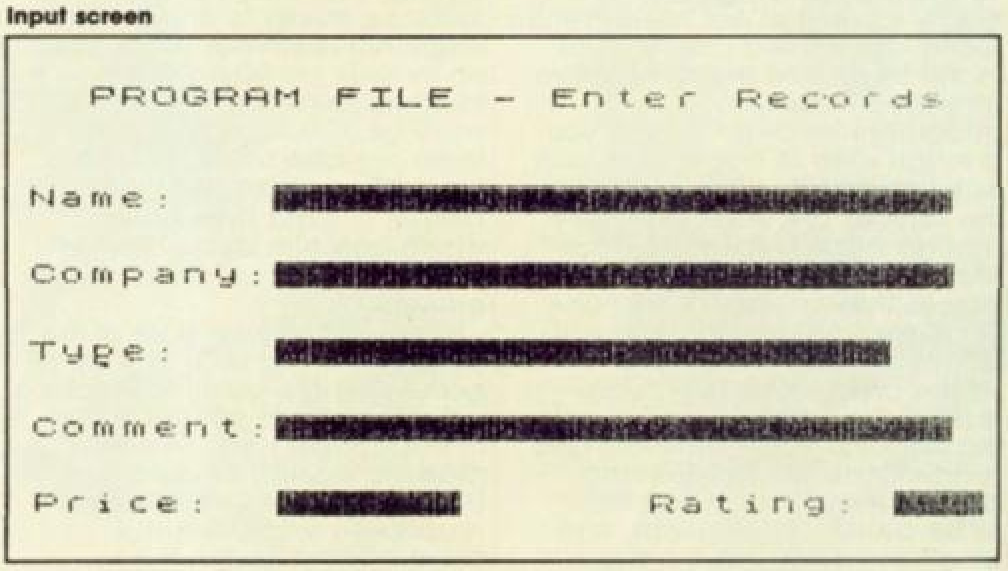
It may be as well to consider the screen handling and cursor position in relation to the way in which the date will be stored, and with this in mind, I think it would be wise to decide on the way the data will be handled.

Data Handling

Perhaps the most obvious way would be to use a three DIMensioned array on the lines of DIM f$(200,6,22), where 200 is the number of records possible, 6 is the number of fields and 22 is the length of the longest field.

With this method pointers 'rec'

Input screen



could keep track of the first element and 'field' the fields under scrutiny.

The problem is that it uses a lot of memory, all the fields are 22 characters long even those which only need 3 and 6 characters. The DIM used takes up 26410 bytes.

It is possible to convert all data of one record to string form and construct a single string with the data in it and store it that way.

What I mean is LET r$=name-field+Company-fields+Type-field+Comment-field+Price-field+Rating-field. The total length will be 94 and if we say DIM f$(200,100) — allowing four extra characters per record — the memory used is 20008 bytes. Quite a saving.

Of course if you wanted to be clever you could always write the data straight to a memory block and SAVE/LOAD it using the CODE function rather than the DATA() system needed with the DIMs.

It could also be useful storing the field lengths in a dimensioned array for reference when checking input and position on screen.

Back to Business..

It may be worth considering using a temporary store during the actual input, something like DIM t$(6,22) at the start of each record entry, and when the entry is complete then transfer it to the main storage format.

Input will almost certainly be by INKEY$ and the key pressed stored to a temporary string for checking. The first check may well be to trap 'no key pressed'.

Perhaps the second check could be for abandon or exit record entry routine.

The cursor position could be used in two variables (x & y in my example) and if the key pressed is the one which invokes movement x or y should be amended as required and then the old cursor deleted and the new one printed. Checks should be made for moving beyond field limits — stop at end or wrap around effect?

Any checks for functions such as delete or enter need to be done before we can finally move on. Once a key press passes all the above then we can assume it is an entry to the field, some validation needs to be done so that letters cannot be entered into a numerical field (but don't forget to allow for a decimal point!) and non alphanumerical characters are prevented from being entered in a "C" field.

If it passes all tests then add it to the field data, move the cursor right by one position (make all the tests for field limits as before) and don't forget to

Set up input screen routine

```
8999 REM screen display
9000 INK 9: PAPER 0: BORDER 0: C
LS : PRINT "  PROGRAM FILE - Ent
er Records"
9010 RESTORE 9991: FOR i=1 TO 6:
 READ x$,y,x,flen,t$: PRINT AT y
,x;x$;AT y,x+8; INVERSE 1;"
                   "( TO flen): IN
VERSE 0: NEXT i
9991 DATA "Name:",4,0,22,"C"
9992 DATA "Company:",7,0,22,"C"
9993 DATA "Comment:",13,0,22,"C"
9994 DATA "Type:",10,0,20,"C"
9995 DATA "Price:",16,0,6,"N"
9996 DATA "Rating:",16,20,3,"N"
```

print the character on the screen.

If it is the final entry in the record then store it all in the file and clear the temporary field store ready for a new set of data — Is it worth an verification that all is OK before finally storing it?

Possibilities

It may be worth using a modular approach, from what I've just said it can be seen that some sections, especially the cursor position needs to be checked on several occasions, on a movement key, on enter, on data entry and on delete. If it was a subroutine then each event could call it.

If the temporary string is used and repositioned then the same routine could be used for an amend option, except that on entry 'rec' would be set and the temporary array filled with the stored data.

4. Further Development.

What a vast area to consider! Obviously you could complete the database program adding the functions Display, Store (to tape, microdrive or disk), Search, Delete, Amend, etc. The program could be adjusted to cover other applications by changing the field headings and field lengths.

In fact a separate program could be written to set up such details and written to a file which the main program could read when run and which would initialise all these features. The program could be made into a universal, general purpose database utility then.

If the routine was carefully written it should be possible to call it for specific input usage, setting up the heading, y pos, x pos and field length and type before calling.

Contact

Don't just sit there, get in touch with us! If you encounter any problems, want to tell me you think I'm taking the wrong approach or just want to say hello, then drop us a line.

This is the most complex Project X that I've set and I'll be interested to see how you rise to the challenge. If you have produced a solution which you think is appropriate, even if it uses a different method than the one I've discussed, tell me, send me a copy.

There are only two things I ask, first that your submission is original work and *unpublished elsewhere* and secondly that you do **not** send a full database program; just the input section and any initialising code that it may need to run a demo.

One of the important features of such a routine is that it should be compact — the less room it occupies then the more records the database can hold.

So why not send us short, compact efficient input routines that can be run. A written or printed listing will do, but a cassette copy does help me considerably. If it is based on the screen display I've given, so much the better (I used it deliberately because it has varying length fields, screen positions and data types!).

I look at all letters and programs sent in, that's why we only set Project X every second month, there is too much to look at to do it properly on a monthly basis! The solution(s) which we consider to be most appropriate will be featured in a future issue and their authors suitably rewarded.

Next month I hope to present a few interesting solutions to the second Project X, the Shuffle, see you then ....

PICASSO

Listing

```
  5 CLEAR 44995: PRINT "LOADING
 CODE": GO SUB 9600
  10 REM %%%% EXTRA CODE %%%%
  20 RESTORE
  30 READ A: IF A>25555 THEN  LE
T START=A: LET BYTE=0: GO TO 30
  40 IF A=999 THEN  GO TO 90
  50 POKE START+BYTE,A: LET BYTE
=BYTE+1: GO TO 30
  55 REM
  60 DATA USR "A",16,16,0,198,0,
16,16
  61 DATA USR "B"+2,0,1
  62 DATA USR "B"+6,0,7
  70 DATA 65100,96,234,0,0,0,2
  80 DATA USR "C",1,223,2,223,1,
253,1,251,3,127
  81 DATA USR "D"+7,16
  90 REM
  95 FOR F=64767 TO 65023 STEP 2
: POKE F,10: POKE F+1,254: NEXT
F
  99 DATA 999
 100 REM %%%% INITIALISE %%%%
 110 POKE 23609,2: POKE 23658,8
 115 PAPER PEEK (USR "B"+6): INK
 PEEK (USR "B"+7): FLASH 0: BRIG
HT 0: OVER 0: INVERSE 0: CLS : P
OKE 23624,PEEK 23693: BEEP .0001
,69
 120 LET L=USR 65025
 130 LET G$="OFF": LET K$="POAQM
": LET Z$="PLOTDRAWCIRCDEL."
 140 POKE USR "B",PEEK 60271: PO
KE USR "B"+1,PEEK 60272
 150 LET MEM=45000: RANDOMIZE ME
M: POKE USR "B"+4,PEEK 23670: PO
KE USR "B"+5,PEEK 23671
 160 LET N=MEM
 170 LET L=USR 60521: LET CURSOR
=60074: LET DEFN=60273: LET STOR
E=60320: LET GET=60332: LET QDRA
W=60344: LET GRID=60550
 180 LET R=1: LET LX=999: LET LY
=0
 190 CLS : LET L=USR STORE
 200 REM %%%% MENU %%%%
 205 POKE N,0: POKE 65102,0
 210 CLS
 220 PRINT AT 0,0;"MENU"
 221 PRINT '"1 EDIT"
 222 PRINT "2 ALTER GRID"
 223 PRINT "3 PRINTOUT"
 224 PRINT "4 VIEW PROGRAM"
 225 PRINT "5 SAVE"
 226 PRINT "6 LOAD"
 227 PRINT "0 NEW"
 230 PRINT AT 0,20;"STATUS";AT 2
,20;"TIME DELAY";AT 3,20;(PEEK (
USR "B"+3)-1)/9
 240 PRINT AT 5,20;"FIRST BYTE";
AT 6,20;MEM;AT 8,20;"LAST BYTE";
AT 9,20;N
 245 PRINT AT 16,20;"DIMENSIONS"
;AT 17,20;PEEK (USR "D"+7);" X "
;PEEK (USR "D"+7)
 250 PLOT 0,164: DRAW 255,0: PLO
T 0,25: DRAW 255,0: PLOT 155,25:
 DRAW 0,150
 255 PRINT AT 20,0;"DPW2 - GRAPH
IC ASSEMBLER               ROBER
T GLAVAS [C]1986 A.D"
 260 IF IN 31>31 THEN  POKE USR
"B"+2,0
```


A drawing utility by Robert Glavas.

Some machine code routines make this stand out from most of the drawing programs we receive. Included are: a synchronized cursor (interrupt driven), fast drawing, Kempston/keyboard operation and user-defined grids.

Type in the listing and SAVE it. Run it and make any necessary corrections then save the final version with SAVE "PICASSO" LINE 1 so that future loadings autorun.

Menu

The program is menu driven and the options are:

1. EDIT

This is the main drawing screen, you control the cursor either by the keyboard (the keys are user definable from option 'R') or a Kempston compatible joystick and interface.

The ENTER key is used to switch between modes. The current state is displayed at the bottom right of the screen and the modes are activated by pressing the fire button or pressing the selected fire key.

The modes are

PLOT: Plots a point on the screen at the current cursor position in the selected INK colour.

DRAW: Draws a line from the last point plotted or previously drawn to, to the present cursor position.

CIRC: Defines the present cursor position as the centre of a circle, then use Left/Right to increase or decrease the Radius. Pressing fire draws the circle or Space exits the routine.

DEL.: Deletes the last thing drawn on the screen.

To return to the menu press the space bar.

2. ALTER GRID

Allows you to specify the grid size.

3. PRINTOUT

Produces a printer copy of either the drawings or the program that will create it. Tested on ZX compatible printers, ZX LPRINT 3 and KEMPSTON interfaces.

4. VIEW PROGRAM

Lists the program commands on the screen.

SAVE

Saves the picture to tape.

6. LOAD

Loads a picture from tape.

O. NEW

Resets the program.

J. JOYSTICK/KEYBOARD

Changes mode of control.

R. REDEFINE KEYS

Set up the control keys to suit yourself.

T. ALTER TIME DELAY

Specifies the cursor speed.

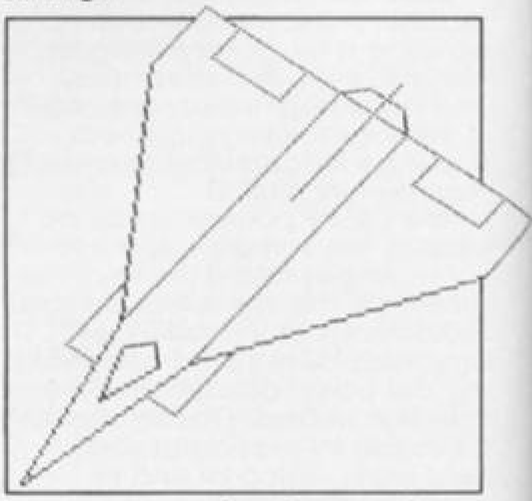
C. CHANGE COLOUR

Allows you to select INK and PAPER colours.

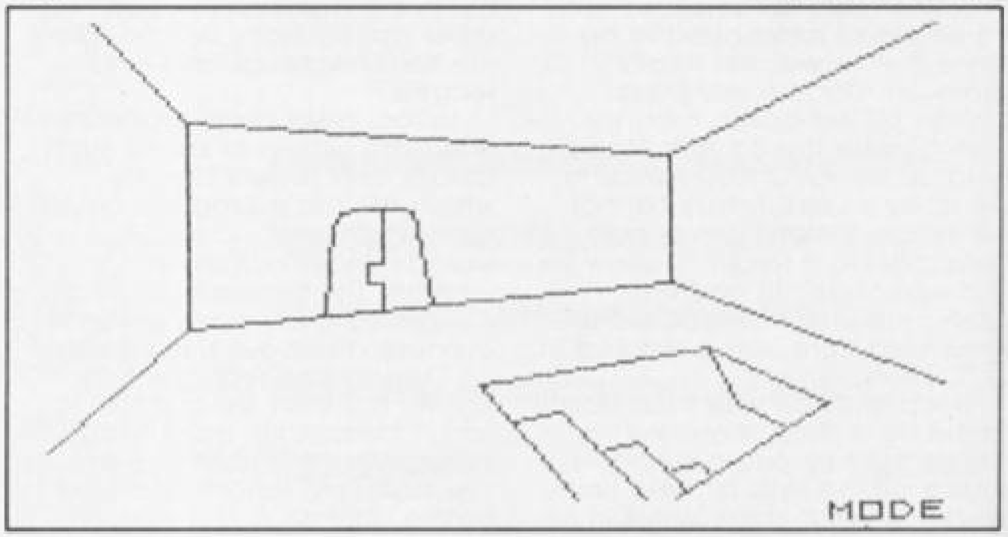
G. GRID ON/OFF

Toggles the grid facility.

There is a display of the status shown on the menu page which will remind you of the existing settings.



Sample screen dumps.



```
270 PRINT AT 11,20;"CONTROL";AT
12,20;("KEMPSTON" AND PEEK (USR
"B"+2)=1)+("KEYBOARD" AND PEEK
(USR "B"+2)=0)
275 PRINT AT 14,20;"GRID ";G$
280 PRINT AT 10,0;"J JOYSTICK/K
EYBOARD"
290 PRINT "R REDEFINE KEYS"
295 PRINT "T ALTER TIME DELAY"
296 PRINT "C CHANGE COLOUR"
297 PRINT "G GRID - ON/OFF"
300 PRINT AT 17,0;"CHOICE : ";
FLASH 1;" "; FLASH 0,
305 IF INKEY$<>"" THEN  GO TO 3
05
310 LET A$=INKEY$: IF A$="" THE
N  GO TO 310
315 PRINT AT 17,9;A$: BEEP .25,
32
320 IF A$="R" THEN  GO TO 500
321 IF A$="T" THEN  GO TO 400
322 IF A$="C" THEN  GO TO 700
323 IF A$="G" THEN  IF G$="OFF"
THEN  LET G$="ON ": GO TO 260
324 IF A$="G" THEN  IF G$="ON "
THEN  LET G$="OFF": GO TO 260
330 IF A$="J" THEN  POKE USR "B
"+2,1-PEEK (USR "B"+2): GO TO 26
0
340 IF A$="0" THEN  INPUT "SURE
? (Y/N):"; LINE S$: LET S$=S$+"
": IF S$(1)="Y" THEN  GO TO 100
350 IF A$="0" THEN  GO TO 200
360 IF A$<"1" OR A$>"6" THEN  G
O TO 300
370 CLS : GO TO VAL A$*1000
380 GO TO 300
400 REM %%%% ALTER DELAY %%%%
410 CLS : INPUT "ENTER TIME DEL
AY (0 TO 9)"; LINE Q$: IF Q$=""
THEN  GO TO 200
411 IF LEN Q$>1 THEN  GO TO 410
420 IF Q$<"0" OR Q$>"9" THEN  G
O TO 410
430 POKE USR "B"+3,(9*VAL Q$)+1
440 GO TO 200
500 REM %%%% DEFINE KEYS %%%%
505 LET K$=""
510 RESTORE 510: DATA "RIGHT","
LEFT","DOWN","UP","FIRE"
520 CLS : PRINT AT 5,0
525 LET A=USR "C"
530 READ A$: PRINT TAB 10;A$;TA
B 18;"?";
540 RANDOMIZE A: POKE DEFN+1,PE
EK 23670: POKE DEFN+2,PEEK 23671
550 IF INKEY$<>"" THEN  GO TO 5
50
555 IF INKEY$="" THEN  GO TO 55
5
560 LET L=USR DEFN: LET A$=INKE
Y$
565 FOR G=1 TO LEN K$: IF K$(G)
=A$ THEN  GO TO 550
570 NEXT G
575 IF A$<"0" OR A$>"9" THEN  I
F A$<"A" OR A$>"Z" THEN  GO TO 5
50
580 PRINT CHR$ 8;A$
585 BEEP .1,30: LET K$=K$+A$
590 LET A=A+2: IF A=65394 THEN
GO TO 600
595 GO TO 530
600 REM
610 PRINT ''TAB 6;"ARE THESE AL
RIGHT ?"
620 IF INKEY$="Y" THEN  GO TO 2
00
630 IF INKEY$="N" THEN  GO TO 5
00
640 GO TO 620
700 REM %%%% ALTER COLOUR %%%%
710 CLS : INPUT "PAPER : "; LIN
E P$: IF P$="" THEN  GO TO 200
715 IF P$(1)<"0" OR P$(1)>"7" O
R LEN P$>1 THEN  GO TO 710
720 INPUT "INK : "; LINE I$: IF
I$="" THEN  GO TO 200
725 IF I$(1)<"0" OR I$(1)>"7" O
R LEN I$>1 THEN  GO TO 720
730 IF P$=I$ THEN  GO TO 710
740 POKE USR "B"+6,VAL P$: POKE
USR "B"+7,VAL I$
750 BORDER VAL P$: PAPER VAL P$
: INK VAL I$: CLS : POKE 23624,P
EEK 23693
760 GO TO 200
1000 REM %%%% EDIT %%%%
1010 LET L=USR GET
1020 IF G$="ON " THEN  LET L=USR
GRID
1070 PRINT £0;AT 0,28;"MODE";AT
1,28; INVERSE 1;Z$( TO 4)
1080 LET L=USR CURSOR
1090 POKE 65102,1
1100 REM %%%% MAIN %%%%
1105 LET X=PEEK USR "B": LET Y=P
EEK (USR "B"+1)
1110 IF Z$(2)<>"R" OR LX=999 OR
N=MEM THEN  GO TO 1170
1130 OVER 1: FOR F=1 TO 2: PLOT
X,Y: DRAW LX-X,LY-Y: NEXT F: OVE
R 0
1150 PRINT £0;AT 0,0;"H:";TAB 5-
LEN (STR$ ABS (LX-X));ABS (LX-X)
;AT 1,0;"V:";TAB 5-LEN (STR$ ABS
(LY-Y));ABS (LY-Y)
1160 GO TO 1200
1170 PRINT £0;AT 0,0;"X:";TAB 5-
LEN (STR$ X);X;AT 1,0;"Y:";TAB 5
-LEN (STR$ Y);Y
1200 REM
1210 IF PEEK 65102=0 THEN  GO SU
B 1700: POKE 65102,1
1220 IF INKEY$=CHR$ 13 THEN  LET
Z$=Z$(5 TO )+Z$( TO 4): PRINT £
0;AT 1,28; INVERSE 1;Z$( TO 4):
BEEP .1,30
1300 IF INKEY$=" " THEN  LET L=U
SR CURSOR: LET L=USR STORE: GO T
O 200
1600 GO TO 1100
1700 REM
1710 IF Z$(1)="P" THEN  GO TO 80
00
1720 IF Z$(2)="E" THEN  GO TO 82
00
1730 IF Z$(1)="D" THEN  GO TO 81
00
1740 IF Z$(1)="C" THEN  GO TO 86
00
1750 RETURN
2000 REM %%%% GRID %%%%
2010 INPUT "ENTER DIMENSION "; L
INE Q$: IF Q$="" THEN  GO TO 200
2020 IF LEN Q$>2 THEN  GO TO 201
0
2025 LET Q$=Q$+" ": IF Q$(2)=" "
THEN  GO TO 2040
2030 IF Q$(1)<"0" OR Q$(1)>"9" O
R Q$(2)<"0" OR Q$(2)>"9" THEN  G
O TO 2010
2040 IF VAL Q$<5 OR VAL Q$>40 TH
EN  GO TO 2010
2050 CLS : POKE USR "D"+7,VAL Q$
: LET L=USR GRID: LET L=USR QDRA
W
2055 IF G$="ON " THEN  LET L=USR
STORE
2060 PRINT £0;AT 0,0;"ALRIGHT? (
Y/N)"
2070 IF INKEY$="Y" THEN  GO TO 2
00
2080 IF INKEY$="N" THEN  GO TO 2
010
2090 GO TO 2070
2999 STOP
3000 REM %%%% PRINTOUT %%%%
3005 IF N=MEM THEN  GO TO 4010
3010 PRINT AT 3,0;"DO YOU WISH F
OR :"
3020 PRINT '"1 A COPY OF THE STR
UCTURE OR"'"2 A LISTING OF THE P
ROGRAM"
3025 PRINT '"space - RETURN TO M
ENU"
3030 LET A$=INKEY$
3040 IF A$="1" THEN  GO TO 3100
3050 IF A$="2" THEN  PRINT £0;AT
0,0;"space - HALT PRINTING": LP
RINT  INVERSE 1;"START": LET OUT
=1: GO SUB 4500: LPRINT  INVERSE
1;"FINISH": GO TO 200
3060 IF A$<>" " THEN  GO TO 3030
3070 GO TO 200
3100 REM %%%% STR PRINTOUT %%%%
3110 LET L=USR GET
3120 COPY
3130 BEEP .5,30: GO TO 200
4000 REM %%%% VIEW PROG %%%%
4010 IF N=MEM THEN  PRINT "A STR
UCTURE HAS YET"'"TO BE ENTERED."
: GO TO 9500
4015 PRINT £0;AT 0,0;"space - EX
IT"
4020 PRINT  INVERSE 1;"START": L
ET OUT=0: GO SUB 4500: PRINT  IN
VERSE 1;"FINISH"
4030 GO TO 9500
4500 REM %%%% DECODE %%%%
4505 LET LN=1
4510 LET F=MEM
4520 IF PEEK F=1 THEN  LET O$="P
LOT "+STR$ (PEEK (F+1))+" , "+ST
R$ (PEEK (F+2))
4530 IF PEEK F=2 THEN  LET O$="D
RAW "+STR$ (PEEK (F+1)-PEEK (F-2
))+" , "+STR$ (PEEK (F+2)-PEEK (
F-1))
4540 IF PEEK F=3 THEN  LET O$="C
IRCLE "+STR$ (PEEK (F+1))+" , "+
STR$ (PEEK (F+2))+" , "+STR$ (PE
EK (F+3)): LET F=F+3
4545 LET F=F+3
4546 IF INKEY$=" " THEN  BEEP .2
,25: RETURN
4550 IF OUT=0 THEN  PRINT O$: LE
T LN=LN+1: IF LN=21 THEN  PRINT
£0;AT 0,0;"space - EXIT"'"other
- CONTINUE": PAUSE 0: LET LN=0:
CLS
4560 IF OUT=1 THEN  LPRINT O$
4570 IF F=N THEN  RETURN
4580 GO TO 4520
5000 REM %%%% SAVE %%%%
5005 IF N=MEM THEN  PRINT "THERE
IS NO STRUCTURE TO SAVE."''': G
O TO 9500
5010 PRINT "WHAT DO YOU WISH TO
CALL THE     STRUCTURE? (10 CHRS
MAX)"
5020 INPUT  LINE A$: IF A$="" OR
LEN A$>10 THEN  GO TO 5020
5030 RANDOMIZE (N-MEM)
5040 POKE MEM-2,PEEK 23670: POKE
MEM-1,PEEK 23671
5050 POKE N,0: SAVE A$CODE MEM-2
,N-MEM+2
5060 PRINT ''"THE STRUCTURE HAS
BEEN SAVED."'"YOU MAY STOP THE T
APE.": GO TO 9500
6000 REM %%%% LOAD %%%%
6010 PRINT "WHAT IS THE STRUCTUR
E CALLED?"
6020 PRINT '"IF THE NAME IS UNKN
OWN JUST     PRESS enter."
6030 INPUT  LINE A$
6040 LOAD A$CODE MEM-2
6050 LET N=MEM+PEEK (MEM-2)+(PEE
K (MEM-1)*256)
6090 CLS : GO SUB 8500: LET L=US
R STORE: GO TO 9500
8000 REM %%%% PLOT %%%%
8010 IF PEEK (N-2)=(PEEK USR "B"
) AND (PEEK (N-1))=(PEEK (USR "B
"+1)) THEN  RETURN
8020 BEEP .2,25: POKE N,1: POKE
N+1,PEEK USR "B": POKE N+2,PEEK
(USR "B"+1): LET N=N+3: PLOT PEE
K USR "B",PEEK (USR "B"+1)
8025 LET LX=PEEK USR "B": LET LY
```

SPECTRUM PROGRAMMING

```
=PEEK (USR "B"+1)
8030 RETURN
8100 REM %%%% DRAW %%%%
8105 IF N=MEM THEN  RETURN
8110 IF PEEK (N-6)=3 OR PEEK (N-
2)=PEEK USR "B" AND PEEK (N-1)=P
EEK (USR "B"+1) THEN  RETURN
8115 LET L=USR CURSOR
8120 PLOT PEEK USR "B",PEEK (USR
 "B"+1)
8125 DRAW PEEK (N-2)-PEEK USR "B
",PEEK (N-1)-PEEK (USR "B"+1)
8130 BEEP .2,25: POKE N,2: POKE
N+1,PEEK USR "B": POKE N+2,PEEK
(USR "B"+1): LET N=N+3
8140 LET LX=PEEK USR "B": LET LY
=PEEK (USR "B"+1)
8150 LET L=USR CURSOR: RETURN
8200 REM %%%% DELETE %%%%
8210 IF N=MEM THEN  BEEP .1,60:
RETURN
8220 IF PEEK (N-6)=3 AND N>3 THE
N  LET N=N-6: LET LX=999: GO TO
8250
8230 LET N=N-3
8240 LET LX=PEEK (N-2): LET LY=P
EEK (N-1)
8250 BEEP .2,20: CLS
8260 GO SUB 8500: LET L=USR CURS
OR
8270 RETURN
8500 REM %%%% SCREEN UPDATE %%%%
8505 IF G$="ON " THEN  LET L=USR
 GRID
8510 POKE N,0: LET L=USR QDRAW
8520 PRINT £0;AT 0,28;"MODE";AT
1,28; INVERSE 1;Z$( TO 4)
8550 RETURN
8600 REM %%%% CIRCLE %%%%
8610 IF X+R>255 OR X-R<0 OR Y+R>
175 OR Y-R<0 THEN  BEEP .1,-30:
RETURN
8630 BEEP .2,25
8635 LET L=USR CURSOR
8640 GO SUB 8650: GO TO 8700
8650 REM %%%% C.CURSORS %%%%
8660 OVER 1: PLOT X+R,Y: PLOT X-
R,Y: PLOT X,Y+R: PLOT X,Y-R: OVE
R 0
8670 RETURN
8699 REM %%%% C.LOOP %%%%
8700 IF INKEY$=" " THEN  GO TO 8
720
8701 IF PEEK (USR "B"+2) THEN  G
O TO 8710
8702 LET A$=INKEY$: PRINT £0;AT
0,8;"R:";R;"   "
8703 IF A$=K$(1) THEN  IF X+R<25
5 AND X-R>0 AND Y+R<175 AND Y-R>
0 THEN  GO SUB 8650: LET R=R+1:
GO SUB 8650: GO TO 8700
8704 IF A$=K$(2) AND R>1 THEN  G
O SUB 8650: LET R=R-1: GO SUB 86
50: GO TO 8700
8705 IF A$=K$(5) THEN  IF PEEK (
N-6)<>3 OR PEEK (N-5)<>X OR PEEK
 (N-4)<>Y OR PEEK (N-3)<>R THEN
 CIRCLE X,Y,R: LET LX=999: BEEP
.2,25: LET L=USR CURSOR: POKE N,
3: POKE N+1,X: POKE N+2,Y: POKE
N+3,R: LET N=N+6: RETURN
8706 GO TO 8700
8710 REM %%%% KEMP %%%%
8712 LET I=IN 31
8713 IF I=1 THEN  IF X+R<255 AND
 X-R>0 AND Y+R<175 AND Y-R>0 THE
N  GO SUB 8650: LET R=R+1: GO SU
B 8650: GO TO 8700
8714 IF I=2 AND R>1 THEN  GO SUB
 8650: LET R=R-1: GO SUB 8650: G
O TO 8700
8715 IF I=16 THEN  IF PEEK (N-6)
<>3 OR PEEK (N-5)<>X OR PEEK (N-
4)<>Y OR PEEK (N-3)<>R THEN  CIR
CLE X,Y,R: LET LX=999: BEEP .2,2
5: LET L=USR CURSOR: POKE N,3: P
OKE N+1,X: POKE N+2,Y: POKE N+3,
R: LET N=N+6: RETURN
8716 GO TO 8700
8720 REM
8725 IF INKEY$=" " THEN  GO TO 8
725
8730 BEEP .2,15: LET L=USR CURSO
R: GO TO 8650
9500 REM %%%% WAIT %%%%
9510 INPUT "": PRINT £0;" PRESS
space TO RETURN TO MENU"
9520 IF INKEY$=" " THEN  GO TO 9
520
9525 IF INKEY$<>" " THEN  GO TO
9525
9530 GO TO 200
9600 REM %%%% MCODE %%%%
9610 RESTORE 9600: LET L=9700
9620 READ A: IF A>30000 THEN  LE
T START=A: LET BYTE=0: LET TOT=0
: LET L=L+1: GO TO 9620
9630 IF A>255 THEN  IF A<>TOT TH
EN  PRINT "ERROR IN LINE ";L: ST
OP
9640 IF A>255 THEN  LET TOT=0: L
ET L=L+1: GO TO 9620
9650 IF A=-999 THEN  RETURN
9660 POKE START+BYTE,A: LET BYTE
=BYTE+1: LET TOT=TOT+A: GO TO 96
20
9699 REM DATA
9700 DATA 60000
9701 DATA 42,111,235,68,77,58,98
,255,254,0,40,5,205,66,235,24,17
73
9702 DATA 3,205,5,235,124,254,17
9,32,1,37,254,2,32,1,36,124,1524
9703 DATA 184,32,3,125,185,200,2
29,205,170,234,225,34,111,235,12
5,198,2495
9704 DATA 3,50,96,255,124,214,3,
50,97,255,205,170,234,58,99,255,
2168
9705 DATA 14,0,71,11,120,254,0,3
2,250,201,237,75,111,235,33,88,1
732
9706 DATA 255,22,8,89,203,126,40
,59,120,254,176,48,54,213,229,19
7,2093
9707 DATA 121,230,7,95,203,57,20
3,57,203,57,62,175,144,71,230,56
,1971
9708 DATA 203,39,203,39,177,111,
120,230,7,103,120,230,192,203,63
,203,2243
9709 DATA 63,203,63,132,198,64,1
03,87,4,175,55,31,16,253,193,174
,1794
9710 DATA 119,225,209,12,203,6,2
1,122,254,0,32,184,75,5,35,125,1
627
9711 DATA 254,96,32,173,201,197,
17,104,255,205,54,235,56,6,125,2
54,2264
9712 DATA 252,40,1,44,205,54,235
,56,6,125,254,253,40,1,45,205,18
16
9713 DATA 54,235,56,1,37,205,54,
235,56,1,36,205,54,235,193,216,1
873
9714 DATA 62,0,50,78,254,201,26,
19,71,26,19,219,254,203,63,16,15
61
9715 DATA 252,201,219,31,203,71,
40,6,125,254,252,40,1,44,219,31,
1989
9716 DATA 203,79,40,6,125,254,25
3,40,1,45,219,31,203,87,40,1,162
7
9717 DATA 37,203,95,40,1,36,203,
103,200,62,0,50,78,254,201,128,1
691
9718 DATA 88,17,104,255,14,1,6,8
,19,62,255,145,103,121,135,79,14
12
9719 DATA 124,18,27,219,254,46,1
,203,71,40,12,203,63,103,44,125,
1553
9720 DATA 254,6,40,6,124,24,240,
125,18,201,16,220,62,0,18,201,15
55
9721 DATA 1,0,24,33,0,64,17,80,1
95,237,176,201,1,0,24,33,1086
9722 DATA 80,195,17,0,64,237,176
,201,42,100,255,126,35,254,0,200
,1982
9723 DATA 254,1,32,11,78,35,70,3
5,229,205,223,34,225,24,236,254,
1946
9724 DATA 2,32,15,78,35,70,35,23
7,67,176,92,229,205,251,235,225,
1984
9725 DATA 24,217,254,3,192,78,35
,70,35,237,67,125,92,126,35,35,1
625
9726 DATA 35,50,176,92,229,205,5
1,236,225,24,192,217,229,217,33,
176,2387
9727 DATA 92,126,111,38,0,229,33
,125,92,126,111,38,0,209,167,235
,1732
9728 DATA 237,82,205,91,236,33,1
77,92,126,111,38,0,229,33,126,92
,1908
9729 DATA 126,111,38,0,209,167,2
35,237,82,205,91,236,205,119,36,
217,2314
9730 DATA 225,217,201,217,229,21
7,33,125,92,126,111,38,0,205,91,
236,2363
9731 DATA 33,126,92,126,111,38,0
,205,91,236,33,176,92,126,111,38
,1634
9732 DATA 0,205,91,236,205,45,35
,217,225,217,201,175,95,203,124,
40,2314
9733 DATA 2,30,255,85,76,71,195,
182,42,17,193,236,21,237,83,54,1
779
9734 DATA 92,20,33,0,61,126,18,2
54,60,32,3,62,126,18,19,35,959
9735 DATA 124,254,64,32,240,201,
58,119,255,87,33,0,0,68,77,229,1
841
9736 DATA 213,205,229,34,209,225
,36,36,124,254,176,32,240,38,0,1
25,2176
9737 DATA 138,111,48,233,33,0,0,
68,77,229,213,205,229,34,209,225
,2052
9738 DATA 44,44,125,254,0,32,240
,46,0,124,138,103,206,80,48,231,
201,1916
9739 DATA 65025
9740 DATA 243,62,253,237,71,237,
94,251,201,245,197,213,229,221,2
29,253,229,33,76,254,78,35,70,35
,126,35,254,2,32,11,253,225,221,
225,225,209,193,241,195,56,0,254
,0,40,231,237,67,53,254,229,197,
205,96,234,193,225,24,218,9306
9800 DATA -999
```

PICASSO

ARKANOID

Welcome to 2nd Generation Breakout.

Arkanoid
Imagine
£7.95

My first reaction when I saw this was to think that Imagine had a bit of a cheek to try and pass off a tarted up version of Breakout as some sort of space age mega-game. But then I've always been a sucker for a good game of Breakout and after a couple of tries I soon found myself hooked.

When you read the instructions on the cassette inlay you get a lot of drivel about your Arkanoid mothership being destroyed and your escape vessel, the Vaus, being hurled into the void. It seems that the way to recreate the mothership is to move the Vaus left and right and bounce a ball against the barriers ahead of you. Once you've got through about thirty levels of this you get to meet the 'dimension changer' who can save you and the mothership from the void.

What a load of rot. As I said, it's just a version of Breakout (based on the coin-op game by Taito), but instead of the simple version of the game that's been floating around for years this version has a number of added features that put a bit of spark back into an old formula.

The game starts with a simple bat and ball screen, in which the barrier is made up of layers of different coloured bricks running horizontally across the top of the screen. But as you start to hit the barrier capsules are released which, if caught, have differing effects on your ship/bat. Some of them slow the ball down, whilst others arm you with lasers, increase the size of the ship, provide an escape onto the next screen and so on. These all add a bit of variety to the action, especially as you can often find yourself darting around the screen trying to decide whether to go for the ball or to risk going after the capsules.

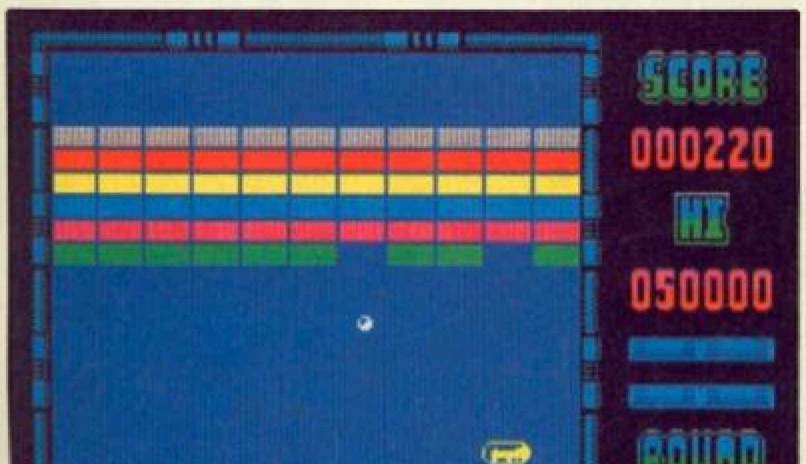
There are also different types of 'aliens' trapped behind the barriers. These aren't deadly, but once they're set free you can bounce the ball off them and rack up a few points.

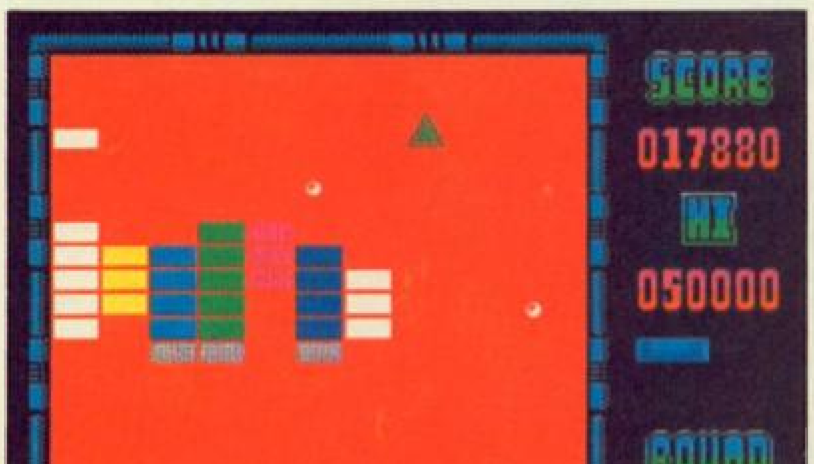
On later screens the shape of the barrier becomes more complex and just to make life even more difficult certain types of bricks are harder to destroy than others, while some are completely indestructible so you'll have a hard time controlling the movement of the ball well enough to find the barrier's weak spots.

There are a couple of minor improvements that could be made; the ship responds quite well to keyboard control, though it sometimes gets a little bit jerky, and I wasn't too keen on using it with a joystick. And the collision detection is a bit suspect at times, as the ball can sometimes go right through the bat when it looked certain that you'd hit it solidly.

It's almost embarrassing to admit, with all the mega-state-of-the-art-arcade-adventures going around, that you can get hooked on something as simple and old-fashioned as Breakout, but after all, it's one of those oldies but goodies that can get you completely hooked, and Arkanoids is currently the game that gets played most in our office. It's only the relatively high price that stops it getting an unreserved monster hit. Eight quid is a bit much for any version of Breakout these days, but Arkanoids is about the best version that you're likely to get.







GAMES

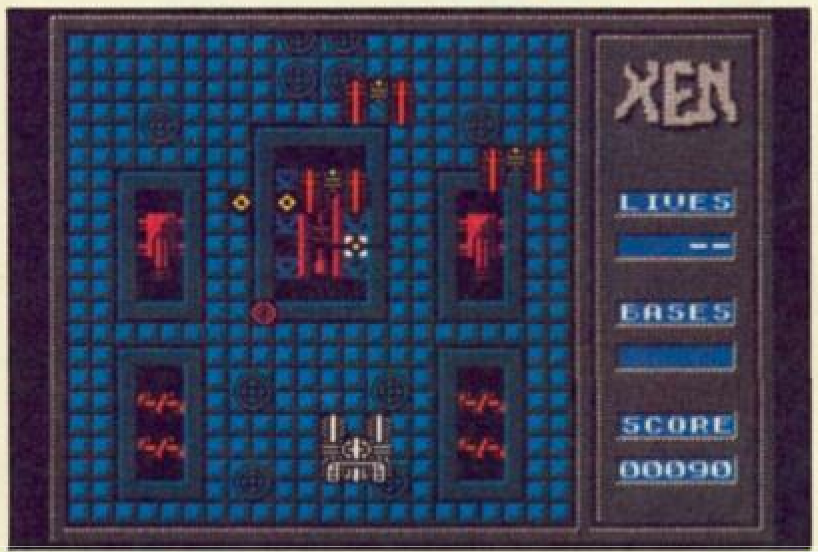
XEN
Creative Sparks
£2.99

After a number of pretty average budget titles, Creative Sparks have come up with a pretty average budget shoot 'em up called Xen. Apart from the fact that they've probably infringed somebody else's copyright somewhere in the packaging of the game, there's not an awful lot to say about the game.

It's one of those games where you control the last surviving craft in the space fleet which defends your home planet, and it's up to you to save the world from the alien fleet. The aliens in this game are called Argons, and they've landed a number of ground installations gathered together in four sectors on the planet's surface. These installations are protected from the air by other alien craft, but you've still got to blast your way through and bomb the lot.

Using reviewer's shorthand I suppose that Xen could best be summed up as a budget Lightforce, unfortunately it isn't in the same league as the original. The prime requisite for any shoot 'em up has to be the speed and responsiveness of control — a shoot 'em up that plods along at a snail's pace is a waste of time, and while Xen isn't *that* slow it isn't really fast enough to build up much excitement. The graphics are all quite blocky, though there is some quite smooth scrolling and animation. There's nothing obviously wrong with Xen, but it's too average to stand out amongst the crowds of other similarly priced titles.







MAD NURSE
Firebird
£1.99

Definitely a game for the dubious taste department. Set in a hospital that's severely understaffed, you play none other than Mad Nurse herself as she tries to stop kamikaze squads of babies coming to harm by swigging down medicine, sticking their fingers in light sockets or crawling to the end of the corridor and plummeting down the lift shaft. You score points by picking up babies and placing them back in their cots. Some babies immediately rebound back onto the floor like gymnasts.

Each level consists of three floors of the hospital and the babies crawl from left to right towards the dreaded lift shaft. The lift connects all the floors and your major decision is spotting which floor is about to become a potential disaster area. Unlike NHS nurses you are provided with 'stun gas' which pauses the screen when things get hectic allowing you time to think out your next life saving move.

The graphics are surprisingly good and if your sense of humour stretches to seeing the funny side of the game you could well enjoy this one. As a game in itself without the babies in jeopardy idea, Mad Nurse would be a little lacking in the action department but when you see those babies heading for the open lift shaft it's difficult not to press the stun gas button and save them. A budget oddity.



TOMB OF SYRINX
Powerhouse
£1.99

Tomb of Syrinx is a sub Jet Willy game in which you are traped in the aforementioned tomb, and have to find the five keys of Syrinx in order to escape. You've got a laser to kill the Guardians (a load of blocky little sprites) with, and you move around the maze simply by moving up/down, left/right, without even the blessing of a jump button to liven things up.

The graphics are state-of-the-art circa 1983, and the maze scrolls jerkily as you move from one room to another. There's a bit of 'House' music included on the tape, and the game crashed thirty seconds into my second game so I had to reload and start all over again.





JUDGE DREDD
Melbourne House
£8.95

You are the one and only Judge Dredd and Mega City One's last chance to stop crime sweeping through the streets. Now at last you can be the 2000 AD comic book hero as you patrol the streets on your lawmaster bike.

You are armed with your "lawgiver" gun that can fire six different projectiles which you use to "arrest" the perps of such evil crimes as murder, tapping (mugging), stookie glanding and tobacco smoking.

A sector of your patrol area of Mega City One warns of any crimes in progress. Move the bike over a crime to rush to the scene. The scene now changes to the ramps, lifts and swamps that form the city which is populated by all sorts of low life. You are here to arrest a murderer, and since gunning down innocent people is frowned on, you'd better get on with the job. Since murder is a minor crime you decide to give the perp a chance and select the Halt response level. This shouts "halt" at the perp when you press the fire button. Warn, the next level up, fires a warning shot but the most common mode used is Kill.

Pressing the enter key swaps between three different displays that appear at the bottom of the screen which are the crime report, informing you of the perps last reported whereabouts, your current location and display showing your response level, bullet type, score and energy gauge.

Finally, you track him down in the rat infested swamp and yell halt! He not only ignores your command he shoots at you. This is no problem to a representative of the Justice Department as you swap to KILL level, select normal bullets and gun him down. They all go quietly, eventually!

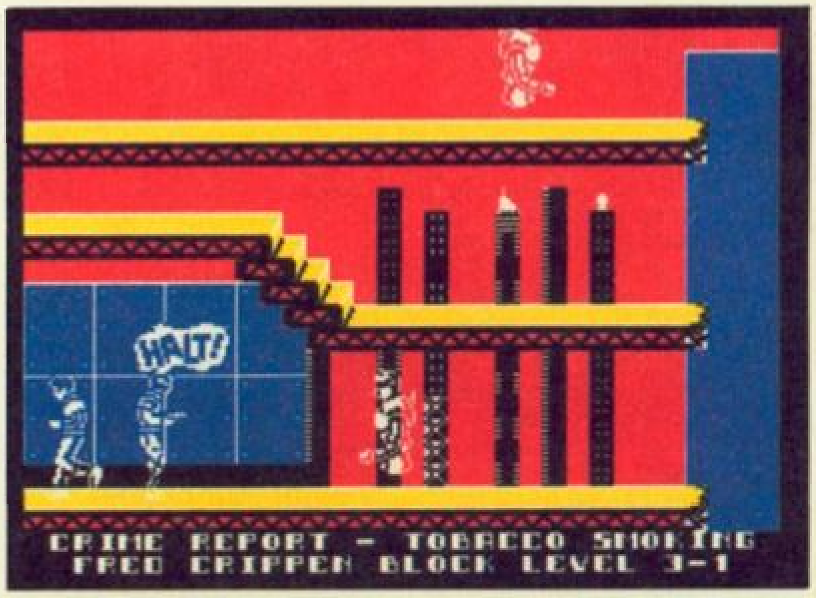
For more persistent perps you can make use of the specialist bullets such as armour piercing, that can fire through the games platforms, incendiary and high explosive that pack an extra punch, ricochet for bouncing around corners (make sure you don't hit yourself) and the heat seekers that always get their man.

Unfortunately, you haven't got time to shoot it out with every perp as you have to get back to the sector map and onto another crime. If you don't you'll lose the game as soon as eight crimes appear in your sector.

You'll soon find out that even Judges can be killed, but luckily this isn't fatal as a replacement is rushed to the sector.

At times it's hard to remember that you're supposed to be the good guy as you can arrest perps by gunning them down. You can also gain bonus points by ridding the city of other vermin such as rats and droids that roam free in the seedy side of the city such as the swamp and the Norman Tebbit block. However, you'll soon know when you've shot something you shouldn't have, as your score will plummet.

The game captures the atmosphere of crime fighting in the future although it has strayed from the comic character which will no doubt annoy the connoisseurs. The rest of us will love it. "Go on perp, make my day!".



MONSTER HIT



TIMEFLIGHT
Powerhouse
£1.99

CRL's relaunched budget label gets off to a less than power-packed start with this dated little shoot 'em up. Timeflight reminds me of a very simple space combat game that I played on a TV games console about six or seven years ago, in which you control a spaceship which flies around collecting objects, and at the same time avoiding/destroying the enemy aircraft that are randomly flying around.

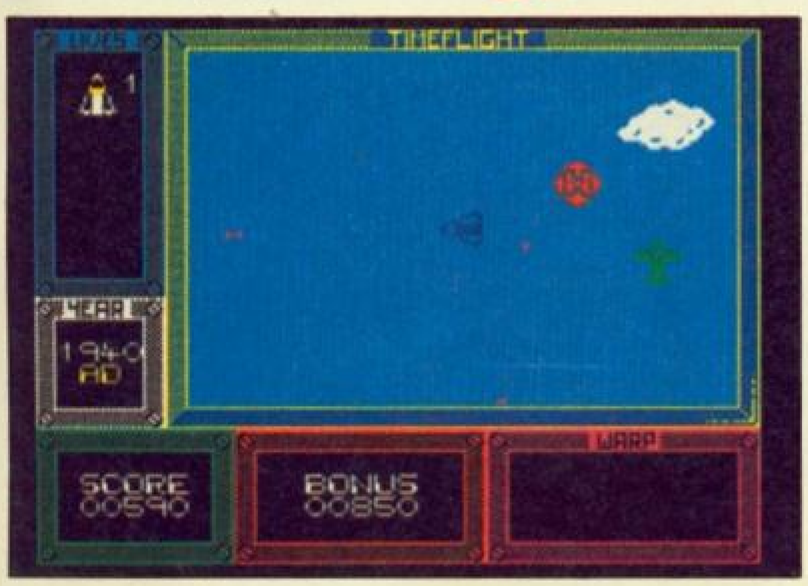
In this case the objects are the four letters which spell out the word 'WARP', and once you've gathered them together you move onto the next time zone which is exactly the same as the first, apart from having a few more enemy ships in it.

Control of your craft consists of rotating left/right and firing little pellets across the screen. The 'time-zone' landscape scrolls under as you move, so that your ship is always positioned more or less in the centre of the screen. On my first attempt I sat there and did nothing, just letting the ship fly up the screen on its own and I got halfway through the second level before I needed to do anything to avoid getting blown up, so the game doesn't exactly score high marks for addictiveness.

For nostalgia buffs only.



GRIM



GREYFELL
Starlight
£9.95

The debut game from Ariolasoft's new label casts you as Norman the Wise cat who must battle the evil Mauron. His friend the Wizard Hitormiss has told him of the Orb of life that Norman must find to free the land from Mauron's evil grip. The game plots Norman's attempt to complete his vital quest.

The screen shows a typical Knight Lore style 3D arcade adventure landscape packed full of trees and boulders to block Norman's path, populated by Mauron's minions to rob our hero of his nine lives and deadly cat eating plants and mines to keep you on your feline toes.

A series of icons beneath the main display can be accessed by pressing the fire button and allow Norman to pick up, drop and examine objects that might help him in his quest, load, save and halt his game position, and choose and fire two different weapons along with an energy and number of lives display.

Unfortunately, as with many other Spectrum arcade adventures the graphics and animation, although impressive, slows down dramatically when there are three or more critters on the screen. Despite this, Greyfell is a fine example of arcade adventuring in which you might have a whisker of a chance to get the bottom of the tail.

GOOD





KOBYASHI NARU
Mastertronic
£1.99

You stand in a closed chamber on the world of Ygor. Ahead of you are three portals that lead to three quests that you must complete to learn the ways of the immortal. Fail and you die!

This is the triple quest of Mastertronic's superb cut price icon driven graphics adventure that introduces a novel new way of adventuring.

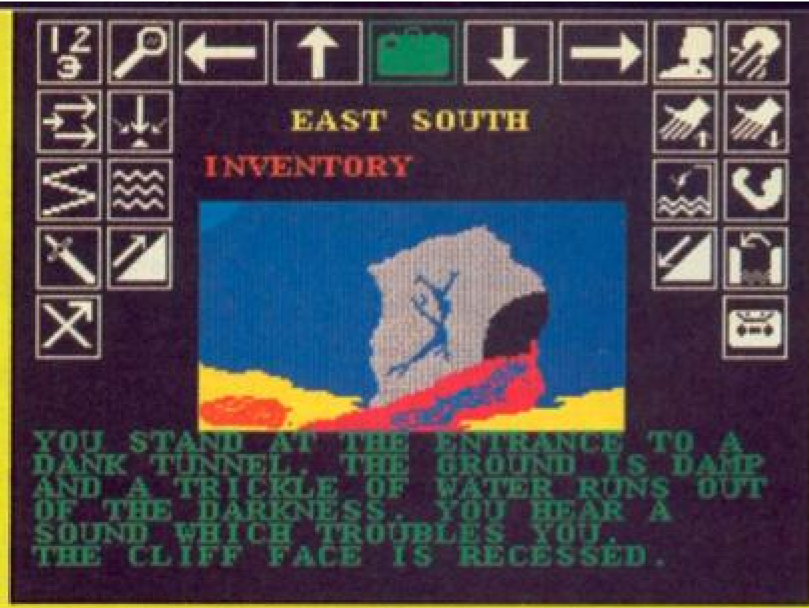
The centre of the screen is filled with a "functional" picture of your current location with a text description below it. Above the picture is a message window through which computer replies to your actions that are performed by selecting one of the 23 icons that allow you to do actions such as get, drop, use, select, examine, analyse, throw, dive, swim and so on. There are also arrow keys to move north, south, east and west and a load/save icon to save your position on tape or in memory (quicksave) in case death lies around the next corner.

Actions such as throw, get, and analyse need further information of what you're going to do it to. Instead of having the frustration of typing words that the computer doesn't understand, you simply use the joystick to pick the word out of the text description! You still get the occasional "you can't do that" message but at least you know which words you can use leaving you to concentrate on the problems.

The three portals are marked knowledge, wisdom and understanding and lead to quests that can be attempted in any order although once you've started one you have to finish it before attempting another. However by using the memory save option you can store your position (where you're stuck) in one quest, quit then try another. If you get nowhere with that one, loading from memory will return you to your original position.

Don't expect to solve the quests in one evening as they're not easy, but mainly consist of trial and error problems, such as how do you get a sword out of a metal block, crawl through a gas tunnel without dying or blowing yourself up, and how to get past a monster that has *tentacles that grow faster than* you can destroy them. These can generally be solved by analysing everything, trying every possible option and a few impossible ones helped by a sprinkling of logical and lateral thinking.

If you succeed the prize is immortality, but is that worth dying for? One thing's for sure, this game is worth every penny.





HOW TO BE A HERO
Mastertronic
£2.99

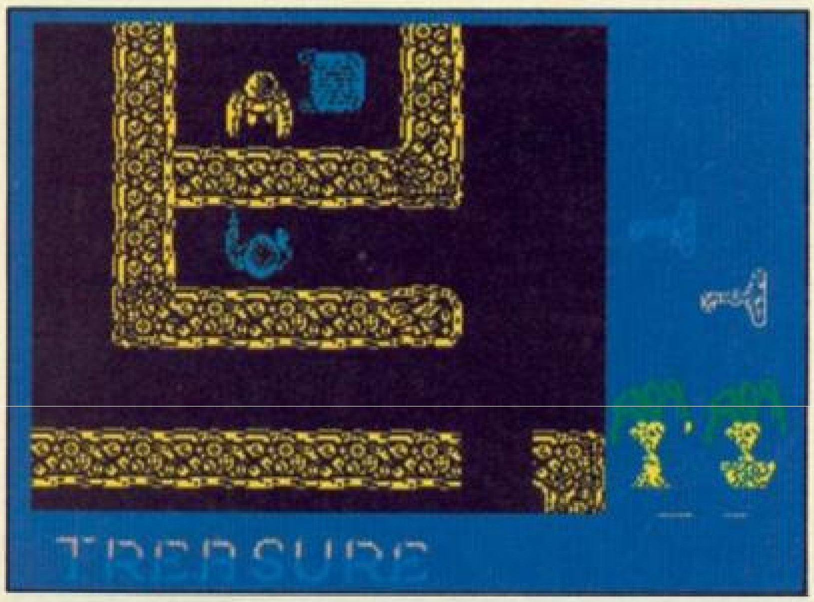
Character building software from Mastertronic's M.A.D. label. How To Be A Hero is actually three games of the wander-around-collect-the-object variety, collected onto one tape.

The idea behind the games is that in each game you can opt for the easy solution, which is to head straight for the maze exit, or you can do it the hard way and attempt to collect the 24 items of treasure hidden in each maze. You don't need to be much of a hero to realise that the treasure is going to be guarded by monsters who will attack and drain off your life energy (illustrated with wacky computer buff humour by two pineapples at the side of the screen). So, scattered around the mazes are additional food supplies to keep you going, as well as the keys that you'll need to get through the locked doors.

The three games are all very similar, and only the differences in layout of each maze separtes one game from another. They all present you with the same sort of overhead view and chunky block graphic, only the minor details vary. In one game you're an explorer in an egyptian tomb, or a space station or a ruined city, and you're always looking for 24 bits of treasure.

They're all passable budget games, and I suppose that getting three of them on one tape is quite good value. Fortunately I'm already such a hero that I don't need to learn how to be one so I shan't be rushing out to buy this.







THRUST II
Firebird
£1.99

Realising that they were onto a good thing with the success of Thrust, Firebird have gone for more of the same with Thrust II.

The basic idea behind this game is the same as that of the original version: you are piloting a small spacecraft which has to collect a number of spherical objects and drop them at a certain point. The combined mass and inertia of the two bodies causes you all sorts of navigational problems, and the main challenge of the game is to master the skills required to steer your way through across the planet's landscape.

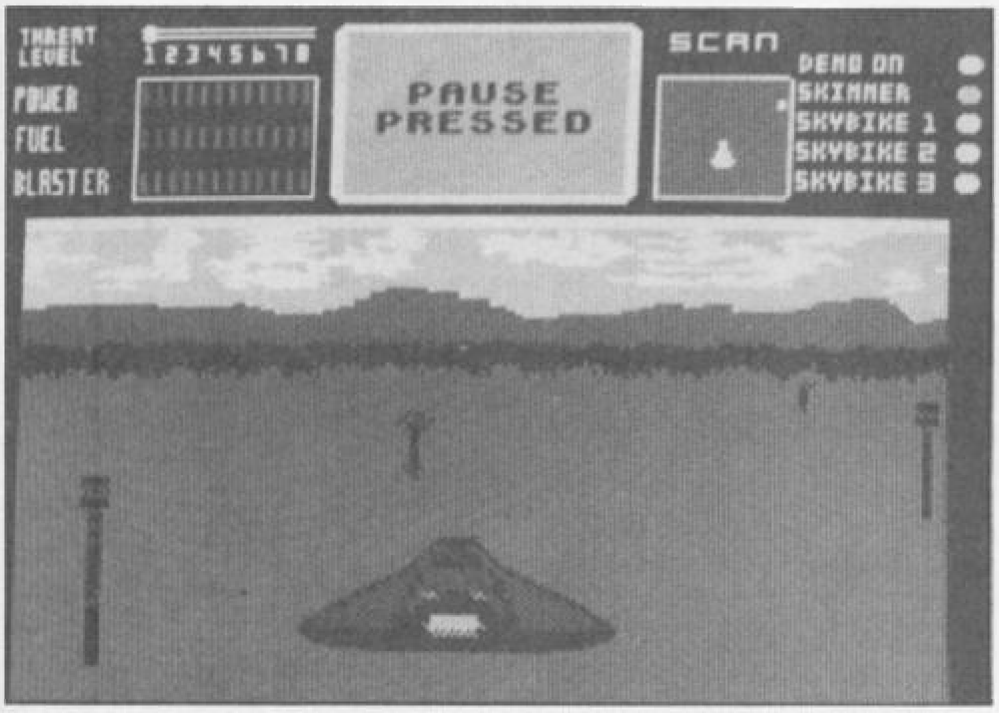
In Thrust II there are one or *two spheres on the surface of* the planet, and these can be picked up quite easily just to get you started. Where the game differs from its predecessor is that the network of underground passages is larger and much more complex, and this time there are also a number of deadly alien robots patrolling the corridors and picking your way past these requires a great deal of concentration. You can pick up chemicals that will destroy these robots, but I'm having a hard enough time with the spheres so I haven't gotten the hang of that bit yet.

Neither of the Thrust games exactly qualifies as a high speed arcade game, but there is something about them which is highly addictive and quite challenging despite the simplicity of the basic idea. There's not that many good full-priced titles around at the moment, so if you're looking for something to tide you over the post-Christmas doldrums, Thrust II could be just the thing.

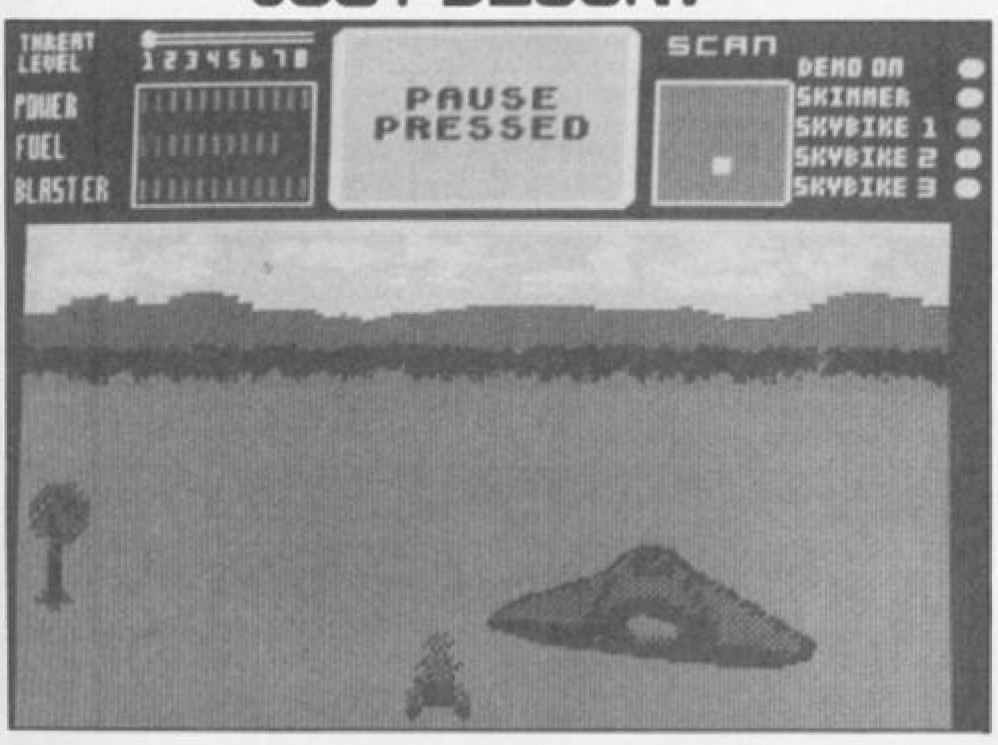




Bust the interplanetary drug ring in Cascades arcade game



...THE FIGHT HAS ONLY JUST BEGUN!



Sky Runner
Cascade Games
£9.95

The plot's rather involved but the action is straightforward shoot em up fare. The inlay blurb rambles on about future civilisations and drug harvesting on distant planets but when you get into the game it turns out to be a pure seek and destroy game.

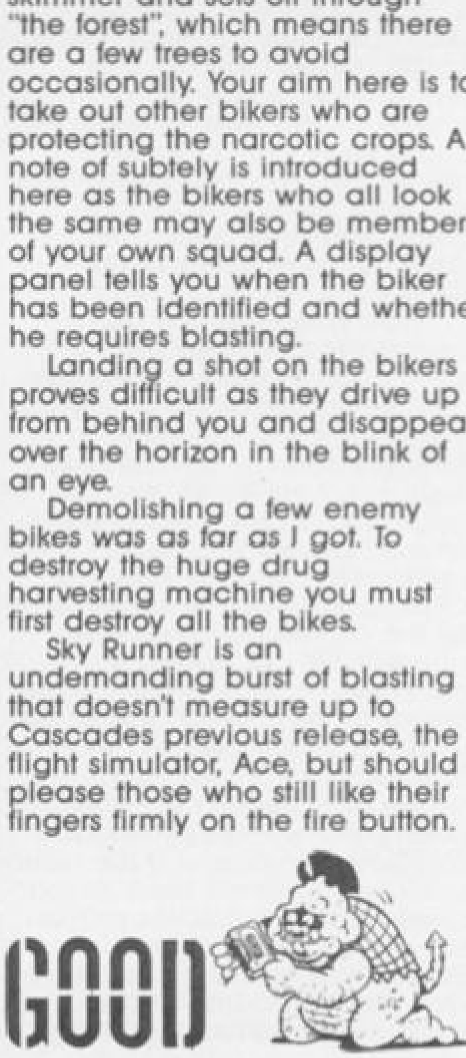
Your mission is to command a two phase operation against the drug growers (the drug is called Sky hence the title). As their plantations are heavily guarded by defence towers, you must first knock them out. Piloting your skimmer craft above the planet surface you have to blast the tops off the towers before they can unleash return fire. The threat level (the number of missiles lobbed in your direction) and speed can be altered quite drastically so that making this an exciting game rests mainly in your own hands. If you play at a slow speed with minimum 'threat' you are likely to doze off but at the other extreme it's virtually unplayable. So it might take you some time to find the right level where it's still a challenge.

When the first phase is complete you are ready to transfer to two wheels. A motorbike is dropped from your skimmer and sets off through "the forest", which means there are a few trees to avoid occasionally. Your aim here is to take out other bikers who are protecting the narcotic crops. A note of subtely is introduced here as the bikers who all look the same may also be members of your own squad. A display panel tells you when the biker has been identified and whether he requires blasting.

Landing a shot on the bikers proves difficult as they drive up from behind you and disappear over the horizon in the blink of an eye.

Demolishing a few enemy bikes was as far as I got. To destroy the huge drug harvesting machine you must first destroy all the bikes.

Sky Runner is an undemanding burst of blasting that doesn't measure up to Cascades previous release, the flight simulator, Ace, but should please those who still like their fingers firmly on the fire button.



my system

David Nowotnik gives a guided tour of his computer set-up and outlines the improvements he's made.

Any one individual's choice of microcomputer and selection of peripheral devices and software is usually just a matter of personal taste and need. I regard myself as an 'enthusiast'; having progressed from ZX81 to ZX Spectrum, I found the sales hype which accompanied the launch of the QL in 1984 irresistable. So, off went my £400 within days of the first announcement, and my QL arrived a few months later, with its EPROM-based operating system precariously hanging from the back of the machine.

I now have an acceptable 'JM' version QL which has given me two years' reliable service. Added along the way were a QL-compatible Microvitec, CUB colour monitor and Epson RS-80 F/T printer. For the package I didn't get much change out of £1,000. Today, by shopping around, you could get a similar system, brand new, for well under £400. Even at today's prices, that represents excellent value for money, especially considering the package of software provided free with the QL.

Last September, the QL seemed to be 'dead', and the temptation to move to another system (noticeable in all my QL-owning friends) was high. Amstrad not only announced their tempting PC 1512 range, but seemed prepared to block anyone developing QL-compatible systems. Now, the CST's 'Thor' seems to be rolling off the production line. New good-quality QL software continues to appear, and there seems to be no shortage of QLs in certain retail chains like 'Dixons'.

The only item I'd want to change now is my printer; the RX-80 cannot produce the near letter quality (NLQ) finish of many more recent dot matrix printers. For certain letters and documents, the RX-80's best typeface just doesn't have a sufficiently 'professional' finish.

Two features of the QL which have always received major criticism are the keyboard and microdrives. The 'feel' of the keyboard is poor, but despite all the pounding that it has received over the past two years (especially when my children play 'Karate'!), my QL keyboard has always been 100% reliable. And, after much use, I have become sufficiently acclimatised to the feel of the keyboard that my typing is no worse on the QL than on more professional keyboards.

Cartridge care

The microdrive system deserved a bad reputation on the Spectrum. But the effect of this was that few professionals were prepared to give the system much of a chance on the QL. I have found the QL's microdrives to be no more troublesome than my experience of 5¼" floppy disc drives. In two years, I've had to throw away only two cartridges. Corrupted files are rare, but, generally these have been successfully rescued with Talent's 'Cartridge Doctor'. The main limitations with microdrives are the access speed, and the maximum 100K storage per cartridge. I'll describe my answer to the speed problem a little later on.

For those who do not share my enthusiasm for microdrives, there are several suppliers of disc systems for the QL. If you go on this route, I would suggest a 1 Mbyte 3½" disc system. In my experience, 3½" drives are more reliable, quieter and faster than 5¼" drives. Single drive systems cost at least £150, with an extra £100 for a second drive. Unless you have a particular requirement for extra storage capacity, faster access or have particularly unreliable microdrives (which could be fixed for around £35), this extra expense cannot really be justified in terms of increased efficiency. You'll be buying one just to spoil yourself!

Having previously used the 1K and 16K ZX81, and 48K Spectrum, what I found to be a surprising limitation on the QL was to have 'only' 128K of RAM. The QL is a particularly greedy machine in terms of memory requirements. For example, the Psion application programs occupy such a large amount of RAM. In 'Quill', documents are limited to around 1,300 words before the whole system slows down to unacceptable levels as the document is partitioned between RAM and microdrive. In the same way, 'Abacus' and 'Archive' have fairly low memory limits placed upon them. For efficiency, extra RAM is essential.

Ram boards

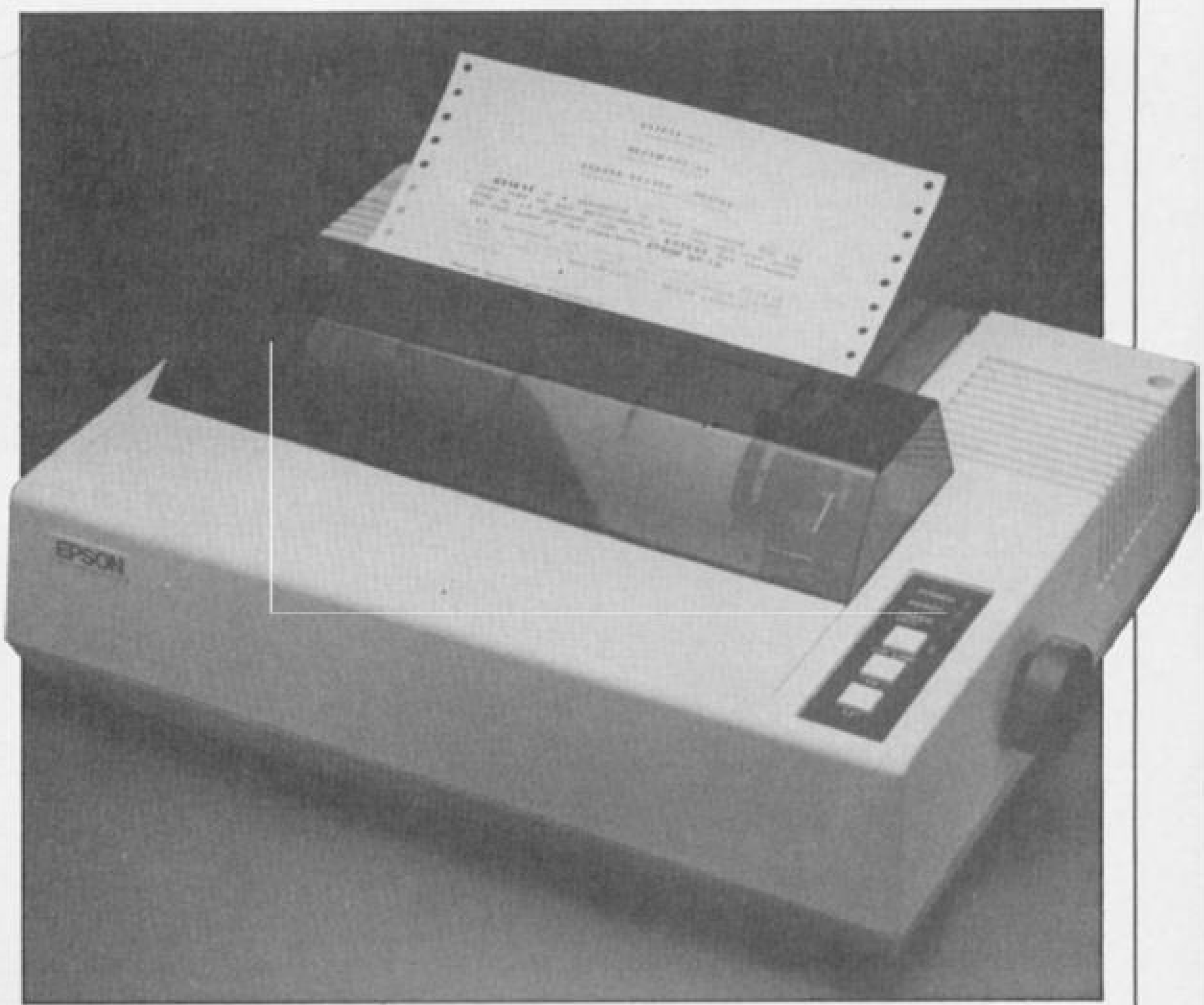
There are a number of suppliers of RAM boards for the QL. I opted for the maximum expansion of a 512K with a 'Thru-Con' board from Sandy (UK) P.C.P. Ltd. At £110 it was marginally the cheapest at time of purchase, and this company was recommended by a friend as providers of good service. And, so far, I have no complaints about the expansion.

The memory board is easily fitted inside the QL through the expansion port to the left of the computer. All that is visible is a connector; there in case I change my mind about a disc system!

One noticeable effect of this extra RAM is the increased operating speed of the QL. External RAM's for the QL use DRAM chips with faster access times than the internal RAM. Another bonus is the ability to utilise the QL's multitasking capabilities a lot better. And a third advantage is the ability to utilise fully something called RAM discs.

For those who have not come across the term before, 'RAM Discs' provide you with a volatile (that is, it will be lost if you turn

off your computer, or you re-set it, or if it crashes) intermediate storage facility which can be used with the same commands as you would use with microdrives and floppy discs. For example, RAM discs can be formatted (FORMAT ram1_200); this allocates space in RAM for RAM discs. Instead of giving the storage medium a name, you usually give a number, which is the number of sectors (1 sector= 512K). Files can be LOADed, SAVEd, COPYed, to and from RAM discs in exactly the same way as microdrive and floppy disc. Except for BASIC programs, major increases in speed are possible. BASIC ASCII files are 'tokenised' during the loading operation; this is the slowest step in loading a BASIC program, and using a storage device with faster access makes little difference to the net speed of transfer.



Qflash

Of the various RAM disc packages available to the QL, I selected 'Qflash' from Eidersoft, priced £18.95. The attraction was the claim to be the fastest RAM disc system for the QL, and a tool kit supplied with this RAM disc which allows all of the contents of one microdrive to be transferred to RAM disc in 7 seconds.

Bearing in mind the screen display of the QL is 32K bytes in size, you'll get an impression of the speed of 'Qflash' RAM discs through a cartoon display. The screen display can be changed 8 times a second; this is achieved with a BASIC program simply using the LBYTES command. At best, microdrives would give you 4 to 7 seconds per frame!

The tool kit with 'Qflash' provides, in particular, two useful extra commands; MDVLOAD transfers all files on a microdrive to RAM disc in 7-14 seconds; FBACKUP transfers all files from RAM disc to microdrive in a similar time. In combination, these two commands provide full microdrive cartridge back-up in just 30 seconds!

I have found the extra RAM and RAM discs particularly useful in the Psion packages. For example, this is how I use them in 'Quill'.

Before loading 'Quill', RAM discs are installed, then the contents of the 'QUILL' microdrive, and a document microdrive are loaded into two RAM discs. For convenience, I have re-written the 'boot' program to do this automatically. As the 'Qflash' tool kit, for security reasons, requires the master to be in mdv2_ during installation of the RAM discs and tool kit, loading everything involves one cartridge change (swapping the 'Qflash' master for a document cartridge), but total loading time is about one minute. I transferred the 'Qflash' RAM disc and tool kit files onto the 'QUILL' cartridge to minimise the wear on the 'Qflash' master.

Using the 'CONFIG_BAS' program on the 'Quill' cartridge, I have set up 'Quill' to expect program files to be on RAM1, and document files to be on RAM2. In this way, when the boot program executes the main 'Quill' machine code program, it does so almost instantly. The same for 'help' files and printer data files.

The advantage to having document files on RAM2 is, again, 'instant' access. Normally, I'll transfer to RAM2 a number of document 'templates' — empty documents with headers, footers, margins, tabs, etc. all set up for particular types of document. For example, I have a letter template, with my address set up on the top right, and 'Dear' a few rows down, waiting for me to add a name.

Also in RAM2 (transferred from microdrive) are any documents which I'll be working on during a session. These will only be the most recent. It's never a good idea to use a microdrive which is close to being full; in this situation, overwriting a document file can get very slow.

Keydefine

Another package which I use with 'Quill' is 'Keydefine' from Psientific software. This is a multitasking program, which sits behind the main application. 'Keydefine' can save a lot of typing of frequently used phrases, sentences, and paragraphs. With 'Keydefine' you can predefine any key to hold text. In practise, say in using 'Quill', pressing ALT and the predefined key will bring 'Keydefine' into operation, the text associated with that key is fed into Quill's input buffer, so will appear on the screen starting at the cursor position. It is possible with 'Keydefine' to inset control characters, to simulate pressing, for example, the function keys, ENTER, and TABULATE.

Back to 'Qflash' for a moment to explain one of my uses of 'Keydefine'. Unlike other RAM discs systems, 'Qflash' does not require you to FORMAT RAM discs before use. Eight RAM discs are available immediately you install the RAM discs. These

'discs' only take up the space of the files you place in them (plus a small overhead); i.e. the size of the RAM discs is variable, depending on their contents, and available RAM. This is a very convenient system as no space is wasted; the rest of RAM is available to QDOS. However, in executing any of the Psion programs, they will grab all available RAM, leaving the 'Qflash' RAM disc with no room to expand. The way around this problem is to create a large dummy file on RAM disc (one program line in the boot program) just before executing 'QUILL', then deleting it once 'quill' is running. So, in every 'Quill' session, my first action has to be to delete the dummy file, to release the space for the RAM discs. Repetitious key presses like this are avoided with 'Keydefine', removing the dummy file is achieved simply by pressing ALT and the predefined key.

Another time saver: Once you are sufficiently familiar with a Psion program, always 'remove' the prompt window at the top of the screen, by pressing 'F2'. You'll be surprised at the difference in it makes, in speed, and in the increase in 'workspace' area you get on the screen.

In producing documents with 'Quill', one limitation I found was to have just one set of printer data codes. For example, three sets of printer data would be required for a draft copy, a final copy on continuous paper, and a final copy on continuous sheets. Currently, I have five sets of printer data on microdrive (which is transferred to RAM1 in the boot program). The trick to install one of these files is to delete the printer_dat file, then BACKUP the required file to ram1_printer_dat. With all these modifications defined for the five function keys, changing the printer data on my system takes a couple of seconds.

My next project for 'Quill' is to write a printer 'spooler'. This will allow a document 'printed' to one of the RAM discs to be sent to ser1 (hence the printer) in the background — that is, a document can be printed out without tying-up the computer; it leaves the user free to continue working with another document while another is being printed. The 'Keydefine' cartridge does contain a 'spooler' program, but this modifies certain control codes, producing odd effects when used with 'Quill' documents.

Another area where RAM discs are having a profound effect is in use with Metacomco's 'C' compiler. As one of my favourite languages, I was very annoyed with this product's slow compilation speed with a standard QL; 10 minutes for only a short 'C' program. With expanded RAM, RAM discs, and repetitive keystrokes 'Keydefine'd, compilation now usually takes under a minute.

Toolkit

Another product I find invaluable is Tony Tebb's Toolkit . Although version II of this toolkit is now available, I don't feel justified in getting that as I have yet to work through all the extra facilities which version 1 provides! Regular readers of ZXC will know that I spend a lot of my time at the QL keyboard programming in BASIC. That work is made a lot easier with Toolkit's full screen editor. I'd feel lost without it!

I hope this description of my QL computer system might provide fellow QL users with a few tips on how they might get more from their machines. If you have any further suggestions I would be interested to hear from you; we'll publish the best of these.

Lin-O-Type

Lin-O-Type
Bradway Software
£8.50.

Linotype is a rather complicated fun program to allow you to use your printer like an electric typewriter to produce small pamphlets, publicity sheets and the like in a variety of typestyles. It is one of the programs of the "Dumpy" family, and, as such, caters for a very wide range of printers and interfaces; morover, since it contains a hefty chunk of BASIC in the Tasword 2 tradition, it can be readily adapted to fit any storage medium: tape, microdrive and discs of various sorts are all catered for. What you get for your money is a master program and a 12 page manual, nicely produced in clear type, photoreduced on a good quality photocopier and neatly bound in stiff card covers. So away you go.

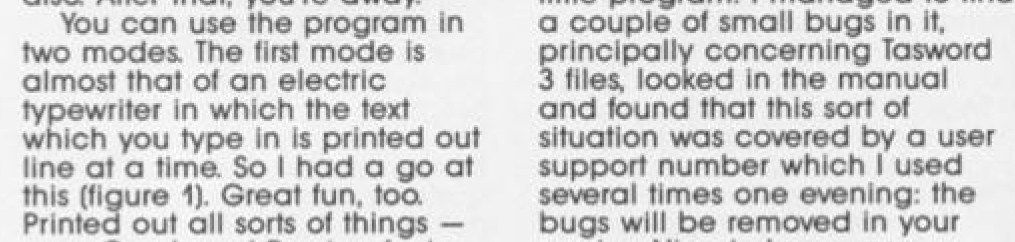
The first step is to use the master program to set up your own customised version. This is necessary because the installation part of the program is subsequently erased to provide more space for use by the Lin-O-Type program proper. The master program goes through a series of questions about your printer and your interface, asking about line feeds and all sorts of other things. I merely selected the Opus Discovery interface, and then just typed in the defaults; this worked a treat on my Epson FX 80. The master program finally ends by saving your customised program on your disc. After that, you're away.

You can use the program in two modes. The first mode is almost that of an electric typewriter in which the text which you type in is printed out line at a time. So I had a go at this (figure 1). Great fun, too. Printed out all sorts of things — even Greek and Russian fonts. Easy! The key to this is the way in which things are changed — press extended mode and immediately a prompt comes up on the input line — "FN?". A glance at the manual and you enter the necessary letter, A to E for the fonts in memory, F to load some more (though you can't load in the middle of a line) and so on. The type style selected actually comes up on screen to help you, so you really see what is happening before the printer gets hold of it.

Emboldened by this success, I read a Tasword file in and watched it being printed out in a variety of styles. Although this is a rather nice little program. I managed to find a couple of small bugs in it, principally concerning Tasword 3 files, looked in the manual and found that this sort of situation was covered by a user support number which I used several times one evening: the bugs will be removed in your copies. Nice to know user support is so good.

All in all, I liked it. The cost of Lin-O-Type on cassette is £8.50, (£1.50 extra for disc, which is not bad), from Bradway Software of 33, Conalan Avenue, Sheffield.

Figure 1. Sample fonts





Strategy amid the shifting sands of the desert in this WWII wargame.

Tobruk
PSS
£9.95

Tobruk

Death in the desert is the theme of PSS's Tobruk strategic tank wargame that also includes an arcade sequence. The game can be played either against human or computer opposition in which you play the Germans and their attempt to take Tobruk. At the time of the attack the allies had only half finished laying a huge minefield. The line, although incomplete gave the Germans an imposing barrier that they must either go round or get their engineers to find a way through. On the other side lay their target, six oases that supplied the allied forces.

As with most wargames Tobruk is played in a series of turns in which the opposing armies move their forces and attack the enemy. Positioning a joystick or keyboard controlled cursor, the identity and strengths of a unit can be displayed which includes a measure of its infantry, artillery, armoured vehicle and support strengths and its mobility factor which determines how many squares it can move through in a single turn. Obviously this is affected by terrain and whether the unit is well supplied. All units on both sides need supplies that provide ammo and reinforcements otherwise they will rapidly weaken. Consequently, Tobruk is as much a battle of support as it is for ground.

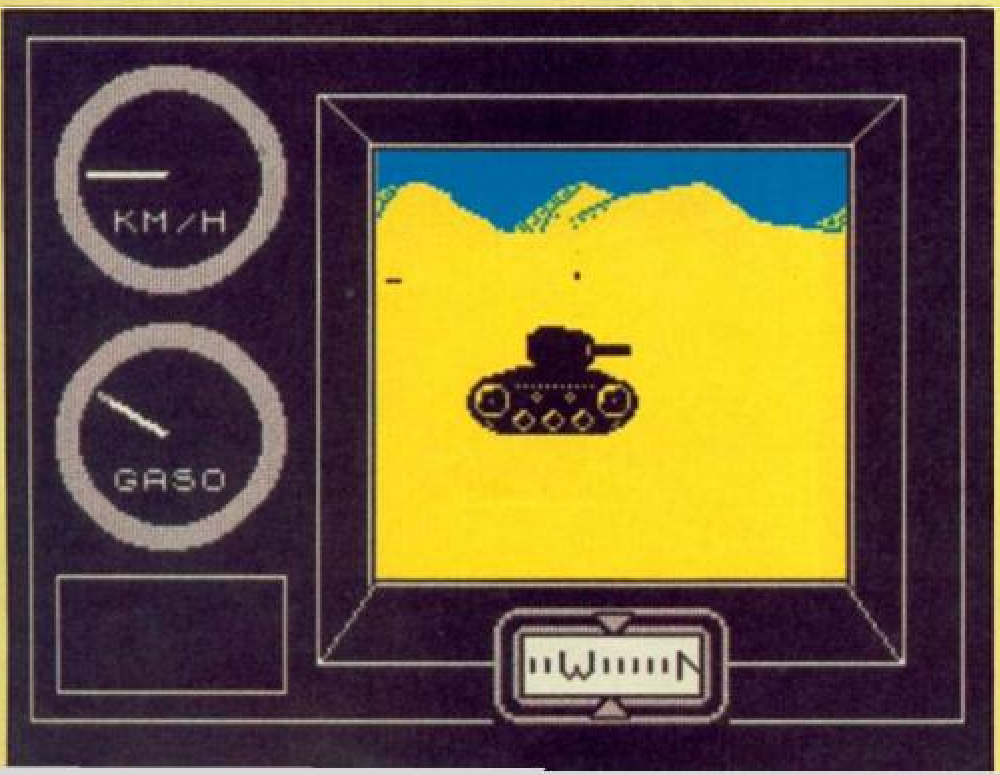
The Allies gain their support from the six target oases and will retreat as these are taken. The Germans have two mobile supply units that can service any unit within six squares. So if you keep our troops together you stand a better chance but be sure you protect the supply trucks or your game will be lost.

Tobruk differs from most wargames with the addition of a vital command phase. This phase allows you to assign your air support and engineers to six different missions. The air support can be assigned to maintaining air superiority, and support ground attacks. Similarly the engineers can be ordered to recover damaged armoured vehicles for repair, sabotaging enemy tanks and mine clearing. Each mission is represented by a coloured box which changes colour as you and the enemy assign support. Red means the enemy's ahead in that mission, yellow indicates that no-one has got advantage but green means you're ahead: If you get a green in ground support then you have an airstrike or free attack against any enemy unit. A green in mine clearing will clear a square of mines. Clear enough and you might catch the allies with a surprise attack.

As with previous PSS Warmaster games, Tobruk also contains an arcade sequence that tests your abilities as a tank commander.

By pushing the joystick you can select between four tank stations in an attempt to blast as many enemy tanks as possible with your allotted ammunition. This system works quite well but it will take some practice before you can swap between the map, machine gun, main turret and driving screens quickly enough to stop your forces taking a beating. Since your performance affects the result of all your battles it's best to avoid this arcade sequence until you've had a chance to practice.

Even without the arcade sequence Tobruk is still one of the best wargames I have ever played and is highly recommended for both gaming rookies and wargaming veterans.





An outstanding naval wargame set in a backwater of history

Legions of Death
Lothlorien
£9.95

LEGIONS

Lothlorien have chosen one of the lesser known war zones of the ancient world to stage their latest military strategy game. This time we are afloat in the Mediterranean sometime between 264BC and 146BC aboard either the Fleet of Rome or Carthage. If you haven't guessed already (and if you have, award yourself a diploma in military history) we are fighting the Punic Wars.





The historical context is that Rome and Carthage were locked in a struggle for the Med and much of this prolonged war centred on the control of the sea channels.

Legions of Death is a one or two player game in which you must ensure that your fleet is victorious in battle and succeeds in protecting its home ports. It is a test of your ability to marshall your seaborne forces over a large area and a bewildering number of situations each calling for a swift command decision. A good touch is added very early in the proceeding when you choose your "conditions of victory", ie the amount of captured gold, territory and destroyed enemy ships that constitute a knock out. This option allows you to vary the length and complexity of the game.

Legions of Death could quite accurately be retitled Legions of Icons. Originally icons were meant to simplify command input but now we've reached the point where there are so many icons to use that things have

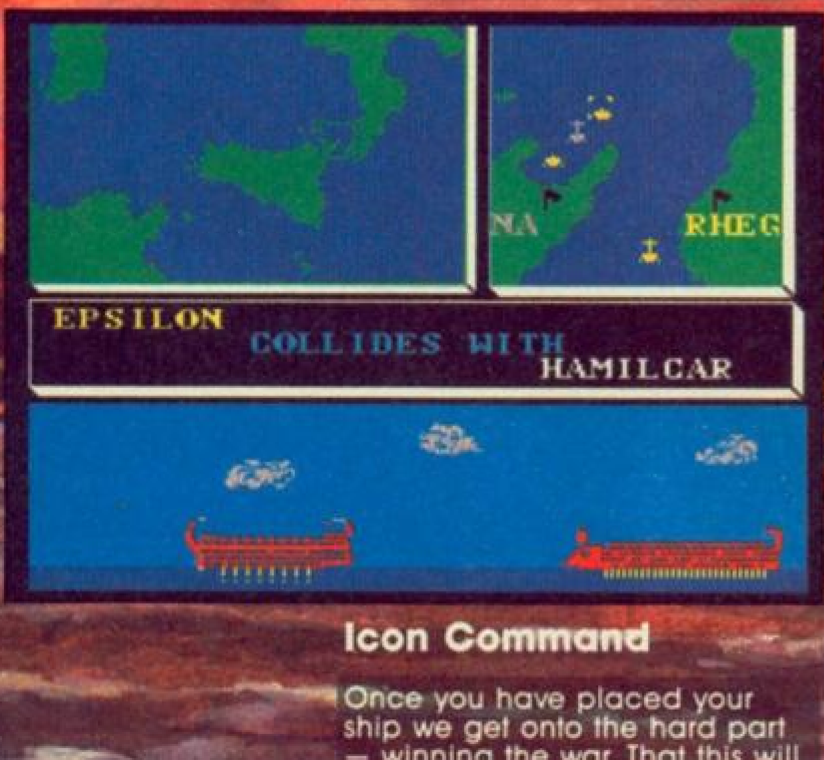
S OF DEATH

become just as complex. But with Legions of Death I'm certainly not complaining as it's a game of great depth and possibilities depending on the degree of commitment from the players.

First off is the fleet building phase where you have 1000 pieces of gold to build, man and equip your ships. There are five types of vessel ranging from the fast but vulnerable Bireme to the slow but formidable Heptares. Selecting modifications to your ships should be thought out in terms of improving your "grappling" capability — the ease with which you can get alongside and board an enemy vessel. One distinctive feature of ships at this time was the Corvus a stout boarding plank that makes boarding more efficient, the disadvantage of course is that it provides an excellent means of entry onto your own ship.



There are so many elements to consider that this initial phase of ship selection can take a good while especially when you are getting used to flipping through the icons. The next stage is the dispersal of your fleet and this is achieved very smoothly on a large scale map of the Med and a close up view of the local area you are in.



Icon Command

Once you have placed your ship we get onto the hard part — winning the war. That this will be no easy matter is indicated by the stern tone of the instruction manual. "To issue orders to your fleet required correct manipulation of the command and enquiry structure of the program". Which means that there are a lot more icons on the horizon and to achieve what you want you have to give orders in a preset way. This icon hopping once mastered will take you to the heart of the game and into the action. When two ships are in combat the bottom third of the screen is given over to an animated sequence of the ensuing battle.

Legions of Death will certainly appeal to wargamers who are looking for something a little out of the ordinary and surprisingly setting it in a relatively little known period gives it a fresher feel rather than replaying battles in yet another WWII or Napoleonic scenario. Through the icon driven system Legions of Death is fairly easy to get into but its complexities become more and more apparent the deeper you get into a campaign. The beginner may find the difficulty level against the computer too daunting but it is a multi faceted game which offers something for strategy gamers of all standards. Well packaged and with an easy to follow manual, Legions of Death should offer hours of enjoyment and might teach you something about the Punic wars as well.

MONSTER HIT

A nightmare scenario of the near future.

Battlefield Germany
PSS
£12.95

Battlefield Germany is a full scale simulation of the war everyone hopes will never happen, a war between NATO and the Warsaw Pact.

According to the background information it took just a year for world events to escalate into a global conflict. Following victories by Iran, Iraq finally crumbled and was followed by Islamic governments in neighbouring Syria, Lybia and Egypt. Israel is next. America responds to Israel's request for aid and sends troops. The Soviet Union objects and an American carrier is mysteriously sunk. As the Superpowers collide, troops build up in Western Europe. 17 June, 1989 the Warsaw Pact launch their offensive.



With one finger hovering over the nuclear "button" you must command the NATO forces to halt the Russian advance or take either side in a two player game.

Be warned, this is definitely an enthusiasts wargame featuring an incredible number of infantry, mechanised infantry, tanks, paratroops, helicopters and even mountain infantry on a massive scrolling map of Central Europe. Luckily, a small map in the top right hand corner of the screen lets you see the way things are going as it gives colour coded territorial map with the units represented by flashing dots. On the main map more detail is available with the forces represented pictorially with further information available when the unit is selected with a cursor.

The game itself is played over either six or twelve turns that can each take about an hour! Each turn consists of two phases, the supply phase (done automatically by the computer) and the action phase when you can move and fight to gain territory.

The computer opponent is tough, with any mistake ruthlessly punished. No quarter is given as the red hordes sweep to victory.

The victory point table that determines the winner and the level of victory depends solely on the expanse of the Russian advance (accepting that it can't be stopped!). Unless you're an accomplished wargamer used to the hexed maps of board games and possess almost total concentration you have little or no chance of winning. Here at last is a real challenge for the experts, beginners should cut their wargaming teeth on Tobruk before they try this one.

BATTLEFIELD GERMANY





Background: PLATOON, Rank Film Distribution Ltd.

FIST II

The fist is back — but does it still have the old punch?

Fist II
Melbourne House
£8.95

Two years ago Melbourne House rocked the world with the first real kung-fu game. Here at last you had total control over the player character and could punch and kick your opponent in a variety of unmentionable places. Soon the shops were swamped with clones and copies of the classic Way of the Exploding Fist. Now Melbourne House has been bought by Mastertronic who hope Fist II will mirror the success of the classic original.

The gametape actually contains two games, Fist II and a practice version which is basically the original Fist but with Fist II backgrounds. When you've practised the kicking, jumping and punching moves you're ready to "continue the legend."

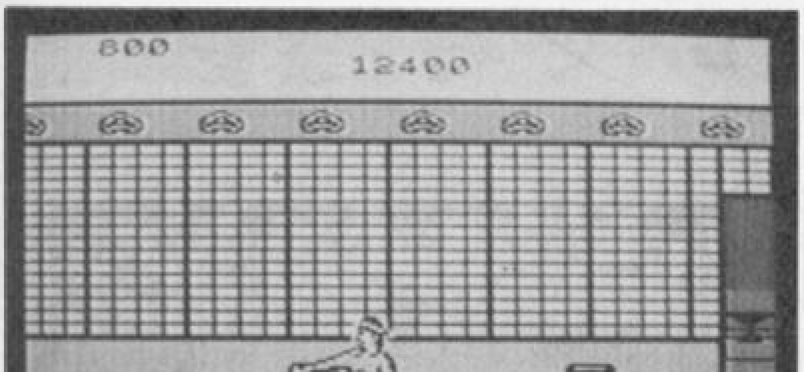
It's now centuries since the time when warriors learned the Way of the Exploding Fist and they've long since been betrayed and the land plunged into the tyranny of an evil War-lord. You play a young descendant of the last Grand Master who must search the land for hidden scrolls and temples to relearn the old skills before you can challenge and defeat the warlord who hides deep in his volcano fortress.

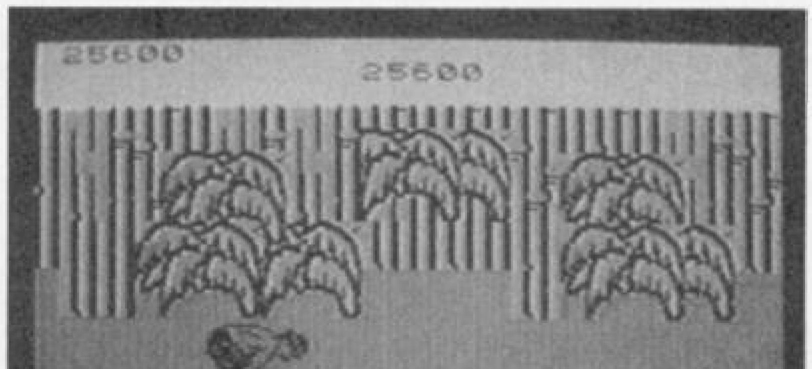
The land is a scrolling landscape full of swamps, caverns, forests and cliffs that are riddled with secret passageways and danger. The danger comes in both human and animal form which you must defeat in kung-fu combat or forfeit one of your three lives. These range from the easy to kill panthers (only problem is they attack in packs) to Ninjas, Assassins and Shoguns that can back up their excellent kung-fu skills by throwing deadly stars at you.

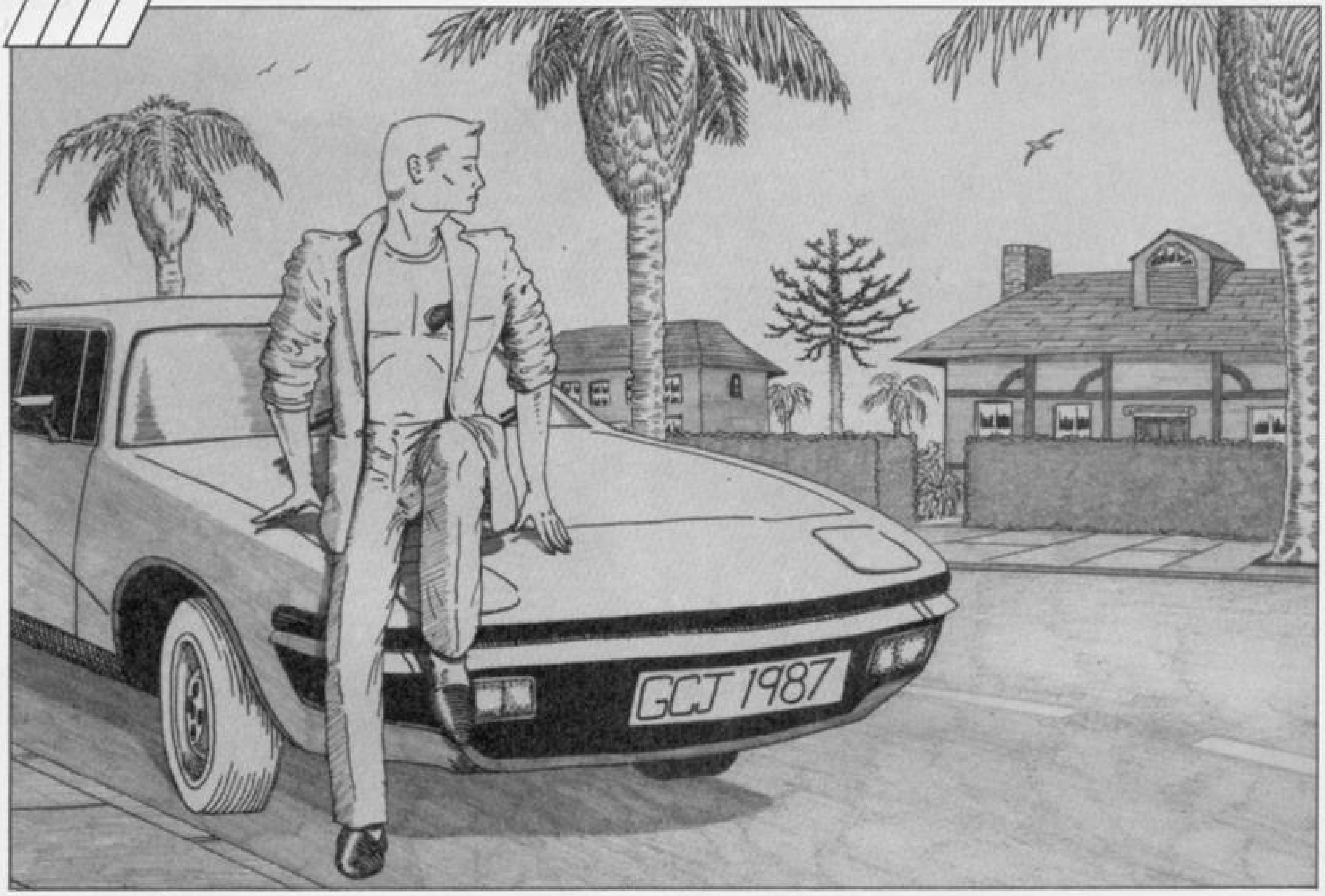
The aim of the game is to collect scrolls and take them to the correct alter to gain the use of a trigram. These are mystical symbols that give you extra powers such as extra strength and healing although some aren't so easy to identify.

Unfortunately you seem to roam through endless screens of nothing that take an age to scroll past. The result is that the action (that's almost identical to the original Fist fights) is diluted, spoiling the game. The trigrams are a brave attempt to add variety to the game, but it will take at least an hour to find the first one. Most people will give up long before then.









MINDPLAY

Peter Sweasey delves into the releases and awards a rare budget adventure Monster Hit.

Little did I expect to receive a detective adventure based in Miami this month. Even less did I expect to receive two detective adventures based in Miami this month! But I did, so this time we feature the budget, Miami Vice inspired Matt Lucas: but unfortunately not CRL's more nautical Murder Off Miami as the copy I received was bugged.

All I was able to play is part one, with no graphics (when graphics appear, the games goes back to the start). What I've seen so far is highly encouraging, so hopefully an unbugged version will be back for review in next month's column.

Mostly budget titles this month, including two from the Players label. Pleasing to see that both are better than the dreadful adventures this Interceptor offshoot debuted with last year. Why not repackage some oldies like Heroes Of Karn while you're at it, Interceptor?

Not long after I started this column, I lamented about the poor state of budget software. Almost a year later, the situation has not greatly improved. Though some encouraging indie product comes in from time to time, the budget big boys are highly disappointing. Firebird only publish Peter Torrance games. I cannot believe that no-one else can write adventures of that quality; in fact, I know there must be far more talented authors reading this very article. Mastertronic persist in refusing to publish Quilled or GACed games. This may at first seem an admirable policy; but when you look at the few adventures they have released, it becomes laughable. There's no reason that Sinbad And The Golden Ship, for example, could not have been Quilled — if it was me, it would have faster responses; and Mastertronic would have saved on their programming costs. US Gold's Americana label seems to have fizzled out before they could release anything more than The Hulk; a shame, because something like Claymourge Castle would make an excellent budget title.

Indeed, there is some great old material for an enterprising budget label to reissue. Like Mikrogen's one-off, superb little adventure, Witch's Cauldron (now Mikrogen is owned by Creative Sparks, perhaps a reissue may occur). Or how about Castle Blackstar, or even some early Delta 4?

On to other matters. Remember I told you about a tape magazine, Spectrum Adventurer, a few months ago? What do you mean, no! Anyway, the latest issue has recently reached me, and there's been a notable improvement. For £2 you get some reviews, numerous hints (including a column from valued Mindplay contributor John Wilson), some news and views, and a mini-adventure. The presentation has improved greatly since I saw it last and the computer format is much better used, with the bulk of the magazine now menu driven. The adventure is mildly diverting for a freebie; but while the text is well written, the vocabulary and responses are painfully limited. Generally recommended, however, for the keener adventurers among you. Try a copy: £2 from SAEC, 4 Kilmartin Lane, Carluke, Lanarkshire ML3 5RT.

THE SERF'S TALE
Players
£1.99

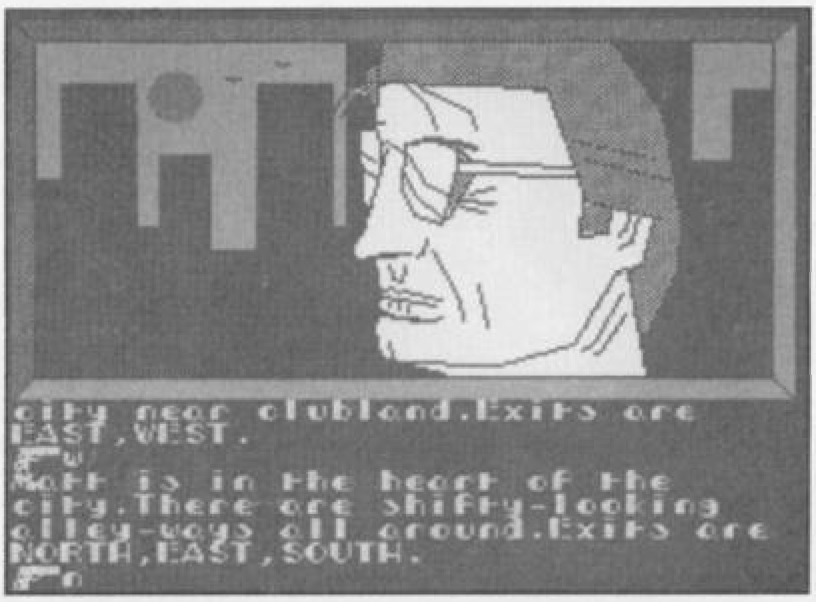
Irrelevant title really — the player's social status has no bearing whatsoever on the game or even the scenario. This tale is a very old story indeed; it's a retelling of the very first adventure! The original mainframe game was never copyrighted, so there have been numerous micro versions, including ones from Melbourne House and Level 9. The puzzles are classics — the bird and snake, Y2, the giant's gate . . .

Is there room for this version then? Resoundingly, yes! Why? Because it's superbly written. The game has actually been improved, with some new twists and locations (the start is much harder than the Level 9 version). The descriptions are atmospheric, well written, quite lengthy and again, not carbon copies. You can perform irrelevant actions like stepping off a cliff or drinking the spring water. Presentation is attractive.

There are some sophisticated features. You can switch between FULL and BRIEF text, a rare but useful function. There is a RAMSAVE/LOAD option — but it's only useable when you find the grimoire which allows you to cast those spells! There's GET and DROP all. You can also PUT things into your pocket and RETRIEVE them, so can carry more.

There's little else that could be done without radically changing the storyline, other than adding graphics perhaps. I prefer this to the Level 9 version — it's friendlier to play and just as well written. The best thing about The Serf's Tale, however, is its price. For two quid, it's unbeatable. If you want to see good, text-only adventuring, this is highly recommended. Unless, that is, you've already solved Adventure, in which case you've been wasting your time reading this review.





MATT LUCAS
Players
£1.99

Matt Lucas is an adventure in the Miami Vice vein. You play the eponymous detective; you have a Ferrari and a boat (though no crocodile). One of the graphics even features a face which, if you squint, looks like Don Johnson. Your colleague Harpinger has been kidnapped; rescue him, and in so doing clear up some of the other vice in the unnamed Florida city. The detection here is not in the Sherlock style; more traditional adventure hazards must be overcome (how to enter a flat, fuel your boat etc).

What is interesting about Matt Lucas is that it is GACed: and GACed well. The other GACed games I've reviewed — including the ones from Incentive themselves — have performed badly in comparison with Quilled adventures. Matt Lucas shows what can be done. Adding a redefined character set and using colour better make much difference, and show up GAC's good points — static graphics, multiple command entry, GET ALL etc. Mind you, I still feel GAC's response time is too slow.

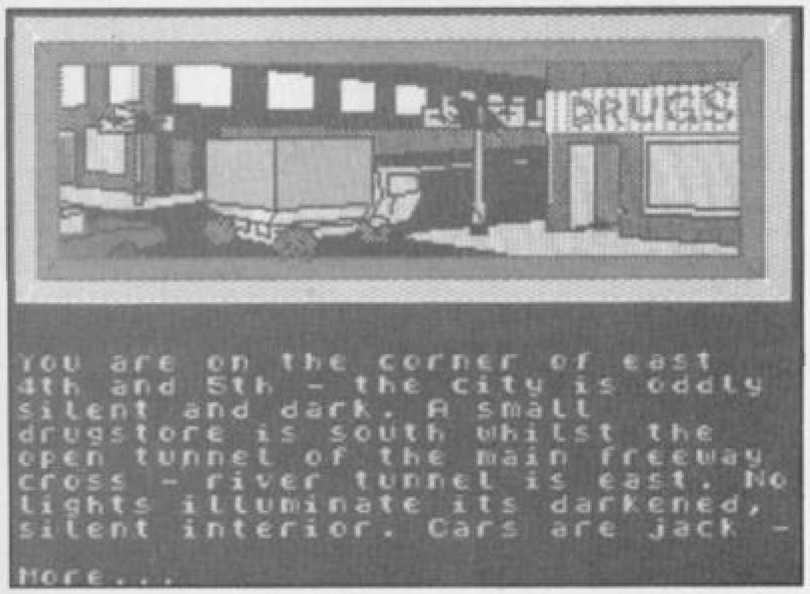
As an adventure, Lucas is *amiable enough, though* generally primitive. There's no true attempt at realism, neither in the portrayal of the locations, nor in the nature of the detective work. Compared with CRL's Murder Off Miami, Lucas becomes laughably crude. But the puzzles are pleasant enough to while some time away with. The vocabulary is, sadly, very small; and the game responses are extremely limited (for example, you can TIP VASE UP to remove a note, but you cannot SMASH or BREAK it).

Matt Lucas is a respectable enough budget title, but nothing special.





FOUR MINUTES TO MIDNIGHT
8th Day
£2.99



8th Day were one of the first companies to see the potential of The Quill for budget software. In 1984 they released six adventures, most of them very good for the time; adventures which I regard with particular affection, since they were among the first I ever reviewed (for the late Home Computing Weekly). Yet — despite the controversial and excellent, full priced HRH last year — 8th Day have never received the success they deserved.

Now the company is back with new versions of those original games. Graphics have been added, and text compressed to allow longer description. Players are offered a choice of graphics or an expanded text version. New features like RAMSAVE/LOAD are included and presentation has been polished. The updated games are being released at the rate of one per month.

Unfortunately, 8th Day have chosen one of the weakest games to start with. The plot: bacteriological warfare has wiped out most of the world's population. You and a few others were immune. Find such people, and together begin a safe new world, de-arm the local nuclear weapons to help this aim.

This scenario was not very well executed to start with, and the game has not been sufficiently revamped. The vocabulary is too tight, and you are far too limited as to what you can do. The graphics are poor, the description uninspiring and the other characters have virtually no life or independence.

Thumbs down for this one, but I look forward to coming games, especially Quaan Tulla and Faerie. I just hope they are updated with more thought.



HELPLINE

Some months ago a veritable flood of letters came through the dungeon grating; whilst in others, like this one, there's only a trickle of mail. Funny old thing, life, is it not? (You are employed for problem solving, not philosophy, thank you — Ed). Mumble. Very well.

Continuing with the multitude of pleas from Simon Fuller that we finished on last time. In **The Mural,** to remove the sandwich eating penguin is quite a task. READ TIMES when carrying the scissors, then type COUPON OUT. Take this coupon to the post box and POST COUPON to receive a Mickey Mouse watch. Head back to the bookcase (best done with the plate), go west and DROP WATCH. Your brownie points will be boosted. In Sherlock, open Basil's safe while he's in London — at about ten on the second morning, for example. At the start of **Return To Eden** (a game forever shamed by the ghastly Australian soap opera which shared its name). DIG DOWN on the mole hill, SLEEP in the comfortable cave you find, to avoid incineration.

Belgian Tom Vandepoel (sorry I spelt your name wrong the other month, Tom) wants to cross the river (deep?) to reach the mountain in Mountains of Ket. I suggest you don't need to. Buy the horse with the money from the Cartographer, then ride east from the village and you'll be at a mountain base.

Mrs Mary Macandriw sent me an extremely entertaining letter in a Department of Employment envelope. She's a Home Economics teacher (proof of her broad profession is the indecipherable way she writes her name, sorry if I've spelt it wrong) and she bribed me with promises of cake for help with Level 9's **Lords Of Time.** You better start baking because here are the answers. To open the door in the inventing room, SAY EUREKA. (In the alcove, think of electricity for the password). Leave the wretch in the stocks alone (justice must be done!) and resist the temptation to throw an apple (unless it's been peeled and cored properly first!).

A. Faulkner is indulging in some budget watersport with **Sinbad And The Golden Ship.** To help him: HIT LAMP WITH AXE to kill the genie. The Glass Key unlocks the door in the mountain, to allow exit from the valley. When you encounter the Ogre and he makes himself invisible, climb the stairs, go west onto the beam, down the rope and east onto the ledge. The beam will give way under the beast's weight. The Djinn will require gems: the green gem from the snake's tongue (TIE ROPE TO TONGUE, PULL ROPE), the blue from the horse and the red from part one. When these three are put in the Idol's rings, a white will be revealed. Take all these, and the flask the idol drops. Put the gems in the flask and your offering is ready. "A" asks why we don't run a telephone helpline. Many answers, including a) It encourages people to give up and ring as soon as they get stuck rather than persevering; b) It takes time to solve some problems — I sometimes have to refer to the game in question; c) The Editor cannot find anyone mug enough to run such a thing, and d) Several adventure clubs, such as Questline and Adventure Probe, run services far better than ZX could.

James Weymouth is stuck in Melbourne House's two classics. In **The Hobbit,** he has reached a dead end in the Misty Mountains. You need to enter a cave (or something — I can't fully remember the name, but it's E, E then N from Rivendell) in the mountainside then WAIT for fate to take its course. In Sherlock, he cannot use the cab. You need to GET INTO CAB (the author has no idea of proper English, not to mention punctuation) then SAY TO CABBIE "GO TO KINGS CROSS ROAD." GET OUT OF CAB at the other end — but the journey will take some time.

Finally, Tim Hills is completely flummoxed by the first part of **Colour Of Money.** Twoflower's been kidnapped, but that's all. You need to do good deeds — with cripples and cats. Also TALK GUARD outside the leaning tower, and SAY WHERE.

As always, my thanks are due to the remarkable John Wilson.

Silicon Surgeon

Loss or discolouration of hair? Unable to sleep? Staring, bloodshot eyes? Are you incapable of holding a conversation except where speech consists entirely of gibberish, along the lines of "if you gave the washing powder to the ogre BEFORE singing the lullaby . . ."? Fingers ever twitching in search of a delete key? Mind ever searching for a synonym for MOVE?

All are symptoms of adventuritis; a sad disease which belongs firmly to modern times. But a cure is at hand. The operation is simple: a pen is taken to hand, the pen is put to the coupon (providing Doctor Bryan remembers to supply it), the pen completes the coupon. The precautions (printed below) are read. The coupon is placed in an envelope, and posted to Mindplay, No 1 Golden Square, London W1R 3AB. The antidote to the dreaded bug — an insoluble problem — arrives, in the form of a solution. Peace and normality prevail.

British correspondents, please enclose a stamped, addressed envelope if you want a personal reply rather than wait some months for the magazine to come out. If you are writing from abroad, just enclose an envelope — I'll add the postage. I try to respond within two months but I can take longer (on the other hand, you might receive an immediate reply). I ONLY DEAL WITH ADVENTURES. Not arcade games; nor technical problems (write to Crosswires about those); nor arcade adventures (Gargoyle games included, not even Heavy on The Magick). Please put the name of the game you're writing about on the back of the envelope. And PLEASE don't write letters asking for general solutions — I just don't have the time to write these out for everyone. (They can be obtained, however, from clubs like Adventure Probe — write to me for details).

See you next time.

Title: ....

Company: ....

Problem: ....

....

....

I can help solve: ....

Name: ....

Address: ....

....

Grange Hill
Argus Press Software
£9.95

Grange Hill.

An animated adventure with the kids from Grange Hill.

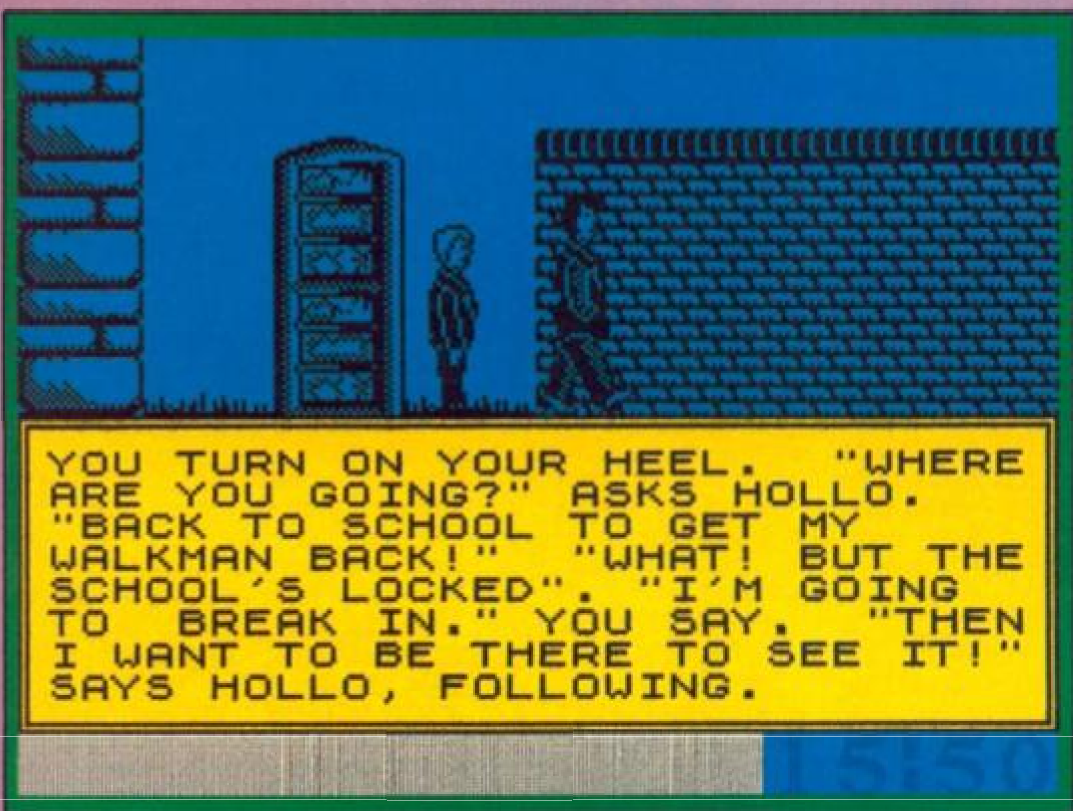
Grange Hill — The Computer Game has been designed as an easy to play adventure with drop down menus and cursor control cutting down the amount of typing instructions required. All you are required to type in is speech when you want to talk to a character or use an object in some way not covered by the verb options.

The plot concerns Gonch and Hollo's attempt to break into Grange Hill after hours and retrieve Gonch's walkman which has been confiscated. Only by getting the walkman back can Gonch avert the wrath of his mum.

Gonch is controlled directly by joystick while Hollo generally just tags along, although at times he needs to be encouraged or reprimanded in order to keep him following you.

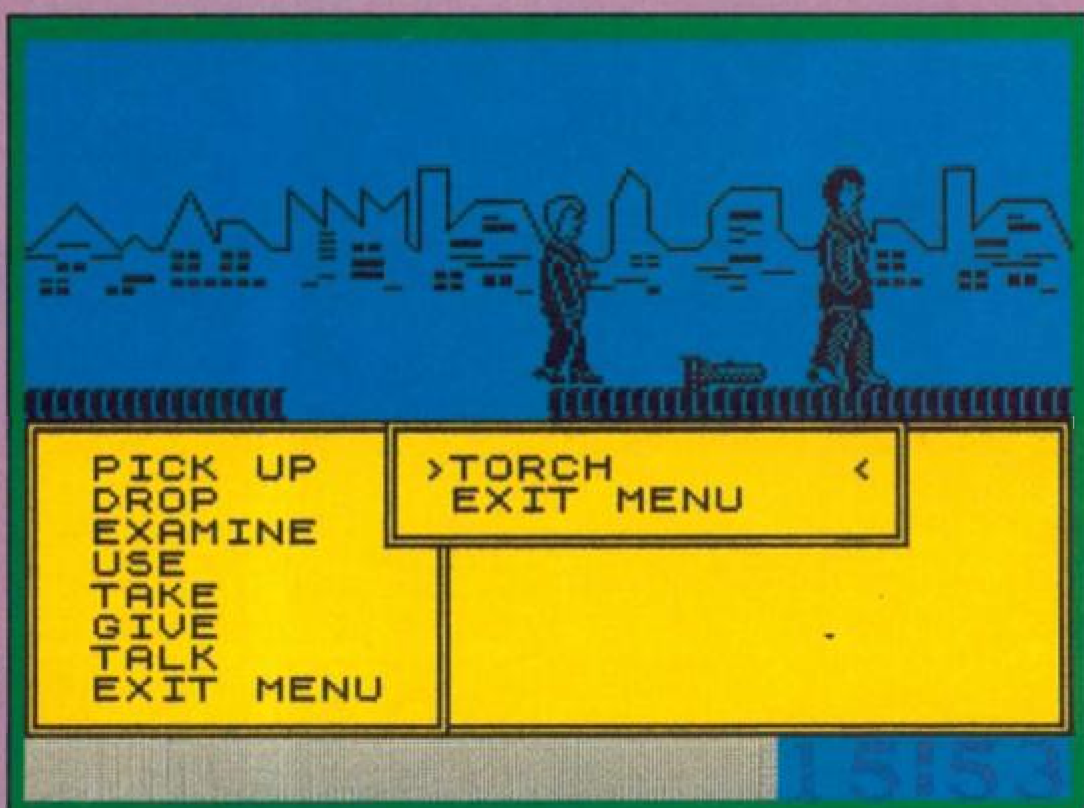
Conventional commands such as pick up, drop, examine, use, take etc are accessed using pull down windows. So with all these commands taken care of what are you supposed to do? The trail to the staff room is strewn with puzzles to solve and hurdles to overcome. As you would expect the objects along the way come into unexpected use later on. By dispensing with the mapping aspect and concentrating on the puzzles Grange Hill proves to be an enjoyable game with a few surprises in store — if for instance you meet "the pusher" and take what he offers, you are instantly out of the game. This is just one of a number of quick exits from the game but you only get caught once.

To add a degree of urgency there is a time limit — you have to get the walkman back to Gonch's house by midnight and the game offers a fairly stiff challenge although seasoned adventurers may find it a bit of a breeze. Obviously aimed at the same age group as Grange Hill viewers, the game with it's easy access system, animated graphics, familiar characters and puzzles will certainly appeal to its target audience.





GREAT



Shockway Rider!

There's nothing pedestrian about gang warfare on the 21st Century walkways.

Shockway Rider
FTL
£7.95

■ It's pleasing to know that nothing ever changes. Here we are, well into the 21st Century and the city's public transport system is still being terrorised by latter day Chelsea and Millwall fans! Of course, the mega-cities don't have tube trains any more, but rather a system of three walkways, moving at different speeds and split up into 12 sections as it circumnavigates the city.

You are a member of the notorious Shockway Riders — the most feared gang in the city. Athletic, aggressive and arrogant as your public relation officer lets everyone know with more than a touch of alliteration. Only one challenge remains. The ultimate one. To go 'Full Circle'. This involves travelling through all 12 zones, battling against rival gang members, the cops, the vigilantes and the much maligned but possibly most dangerous opponent of them all — the general public!

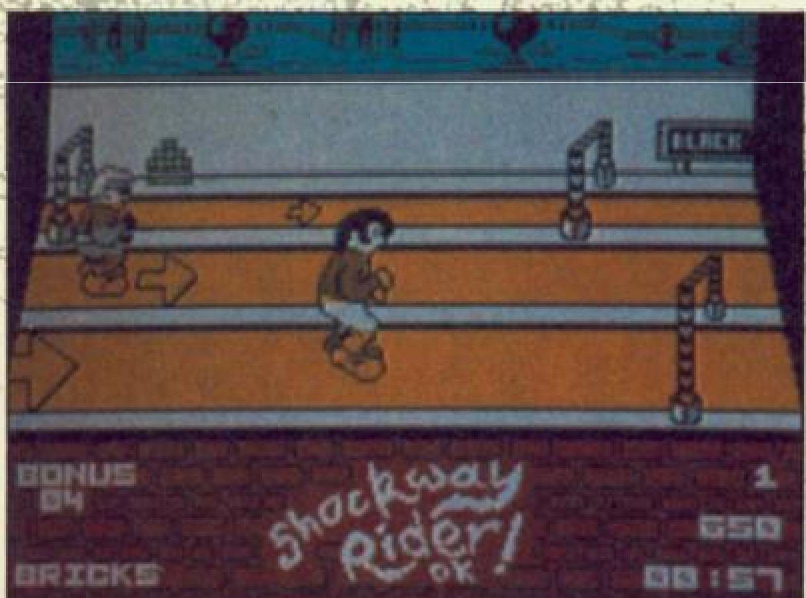
Naturally, you need some form of assistance en route, so it seems wise to help yourself from one of the many piles of bricks casually lying around. These appear to be magic bricks as your supply is unlimited within that district. The advantages of being armed with missiles are two-fold. First, you can take the opportunity to take a pot shot at the various targets that surround your route. Secondly, you can eliminate a fellow traveller/thug from a distance without having to resort to an ungentlemanly display of fisticuffs.

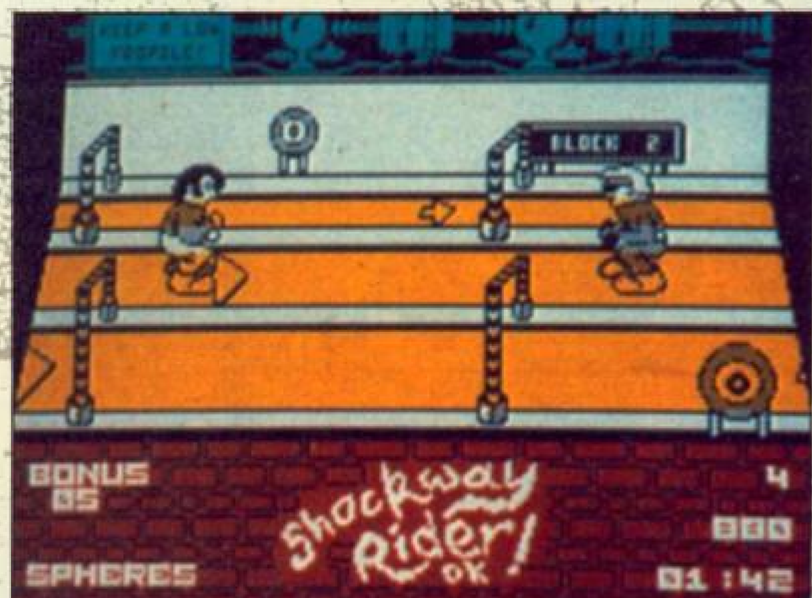
There are other hazards too. As you progress through the levels, barriers block one or more of the walkways. These must be dodged as they are inconveniently placed at head height and the ensuing decapitation is not a pretty sight. Fortunately, this is the 21st Century and your head will regenerate, but only five times!

You can leap about at will on the walkways but hanging back is not a particularly bright idea as you leave yourself open to all sorts of sneak attacks from off screen where you can't see your adversary approaching. Any one who is any one will of course want to shun the slow moving lanes and this is reflected in the points awarded. There is also a bonus for completing a section quickly.

In order to get the feel of what's going on, there is the chance to practise any of the 12 sections before you start your journey. The game is ridiculously simple in concept but highly addictive to play. The black and white cartoon style graphics work well and the gameplay is spot on. This is FTL's second major release after Lightforce (an old office favourite) and they are showing that they are rapidly becoming a force to be reckoned with.







DRAGON'S LAIR

PART II

Dirk rides again in Escape from Singe's Castle.

Dragon's Lair Part II
Software Projects
£7.95

Two games from Software Projects within the last six months, that must be a record. Mind you, Escape from Singe's Castle isn't a completely original title as it's the follow-up to their recent hit conversion, Dragon's Lair. The original coin-op game was too complex to fit into the confines of a little home micro, so Software Projects split the game into two halves (and, presumably, got two games out of one licensing deal).

Having rescued Princess Daphne and despatched the dragon last time around, your character, Dirk the Daring, has returned to the castle which is now controlled by the old Lizard King, in an attempt to find the pot of gold hidden in the depths of the dungeons. But, as in the previous game, you have to guide Dirk past a number of deadly obstacles in order to reach your goal and then escape to safety.

I have to admit that after all the hubbub died down I was a bit disappointed in Dragon's Lair mainly because I found it so hard to play that I simply couldn't get any distance into the game. I eventually gave up because I got bored with always dying on the first or second screens. Part II, however, proved to be a bit easier to play.

The challenges that Dirk has to face in this game include Ye Whirlpools and Ye Rapids, ye amazing Magical Flying Horse, ye olde Mud Monster, and of course the old Lizard King.

I could be wrong but it looks like they've improved the graphics a bit for this game as well. I remember Dirk the Daring looking like Dirk the Smudgy Black Blob in the last game, but here there's a bit more detail in both his figure and in some sort of the location graphics.

Controlling Dirk is a relatively simple matter. Depending on which screen you're on, you use a combination of up/down, left/right, fire/jump to perform the required actions. I wasn't that keen on the joystick control, but using the keyboard was quite simple since this isn't the sort of game where you need 150 fingers to handle the controls.

Dragon's Lair was an innovation in coin-op games because of the way it used a laser disc to store the cartoon-style graphics. But the drawback to this system was that the options available to you were limited because of the vast amount of memory taken up by the high-quality graphics. This meant that, at times, getting past an obstacle was simply a matter of trial and error rather than real skill, and once you'd worked out the correct sequence of moves there wasn't that much left for you to do. However, at its best moments the game also allowed you a bit more scope to approach the obstacles in your own way, and this is where the conversion of Lair II seems to have improved over the first attempt. Perhaps Software Projects saved all the best screens for this game, or maybe I'm just getting mellow in my old age, but it certainly seems like there's more game to play here. Essentially you're still limited to making the right move at the right time, but the 'feel' of the game has improved, and I felt a bit more involved with this game than I did with Lair I, as there's a random element in some of the screens that makes you feel you're in control of events instead of just following a fixed sequence.

So, if you're feeling brave and in the mood for a spot of derring-do, just jam a saucepan on your head, rehearse a few 'Varlets!' and load up Escape from Singe's Castle (and count yourself lucky that the Spectrum version doesn't cost 20p a go).







Escape from Singe's Castle

Wargames Offer

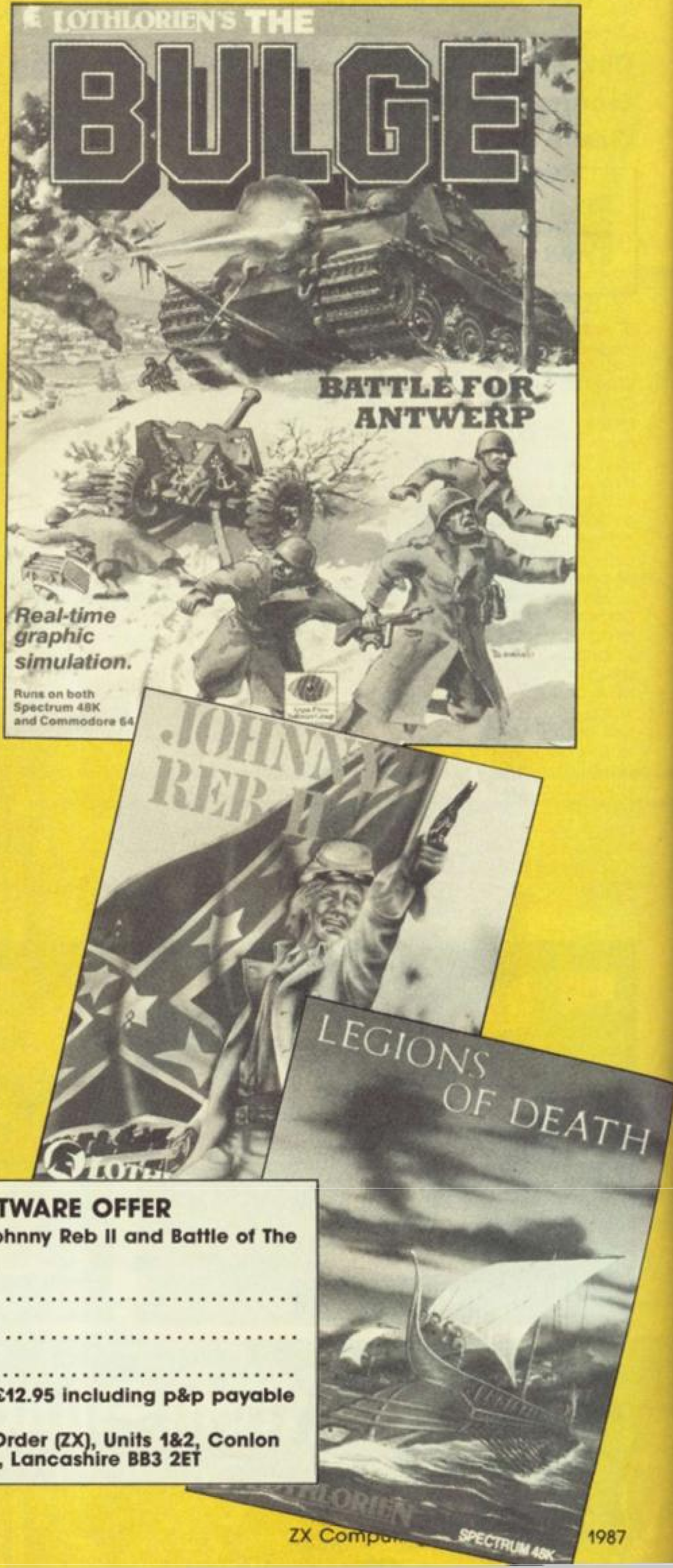
Lothlorien's Monster Hit rated Legions of Death plus two other wargames for just £12.95

Armchair generals get the chance to fight three very different campaigns at a special discount price in the second offer arranged in conjunction with Argus Press Software.

All the games are on the Lothlorien label and pride of place goes to **Legions of Death**, the naval strategy game set in the days of conflict between Ancient Rome and Carthage. Legions has only just been released at £9.95 and is reviewed elsewhere in this issue. With this offer you can pick it up along with two other games for £12.95. **Johnny Reb II** is the recent follow up to Johnny Reb and is an icon driven strategy game set in the American Civil War. It's a one or two player game with three levels of play that centres on the taking/holding of a tactically important bridge. When reviewed in ZX (December) our reviewer stated, "Johnny Reb II is the sort of Wargame that may persuade those who up to now have been left cold by the idea of wargames to think again. For those who are already avid wargamers, this one is a must."

The third game in the pack is **Battle of The Bulge**, a one or two player game reconstructing one of the most important campaigns in the Second World War.

These three games are only available at this special price of £12.95 (incl p&p) by mail order. Do not send your cheques or postal orders to the ZX address but to the address on the coupon on this page. Make cheques and postal orders payable to **Argus Press Software.**



ARGUS PRESS SOFTWARE OFFER

Please send me Legions of Death, Johnny Reb II and Battle of The Bulge.

Name: ....

Address: ....

....

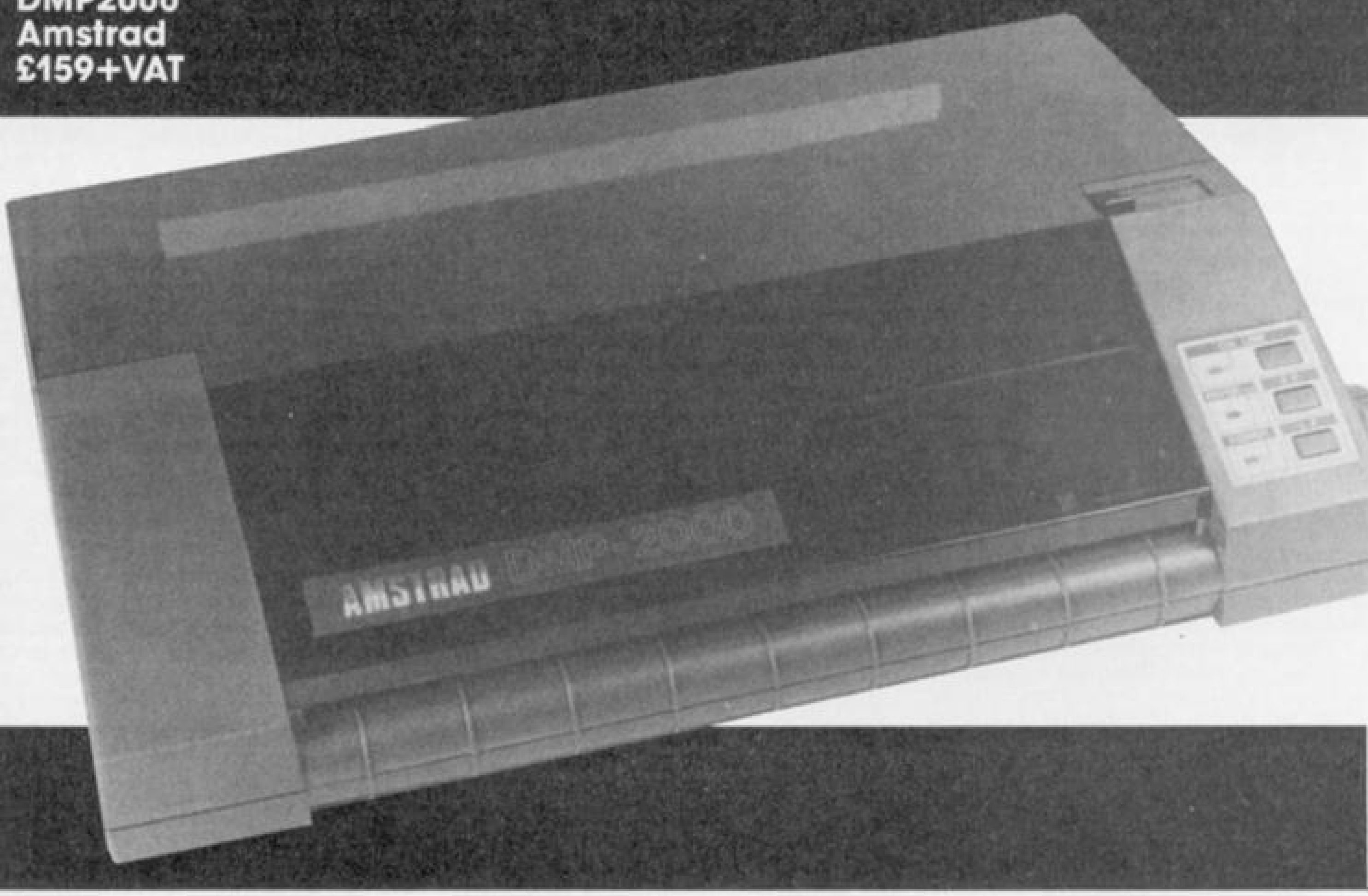
I enclose a cheque/postal order for £12.95 including p&p payable to Argus Press Software.

Send your remittance to: **Argus Mail Order (ZX), Units 1&2, Conlon Developments, Watery Lane, Darwen, Lancashire BB3 2ET**

DMP 2000

A look at the Amstrad strategy applied to printers

DMP2000
Amstrad
£159+VAT



The Amstrad DMP2000 has been getting some good reports lately, so, now that the cheaper printer options (the Alphacom 32 and ZXPrinter) have more or less vanished, a good, cheap(ish) dot-matrix printer is always worth looking at.

Like most Amstrad products it's nothing special from a technical point of view, but manages to provide all the basic facilities you'd expect at a very competitive price.

Freed from its blocks of polystyrene packaging it's an odd looking beast — it looks more like a lawn mower without a handle than a normal printer, and this caused a couple of problems initially as we couldn't figure out how to get the ribbon attached or how to feed paper in. After a bit of office brainstorming we realised what was causing the problem: the machine is 'backwards'. Unlike any other printer that I've played with the DMP2000 has paper fed in from the front and the print head is tucked away behind the paper feed mechanism, so that printing is done as the paper comes out at the back. This makes absolutely no difference to the print quality of course but make the machine a bit demanding of desk space.

All the options

The options available are about what you'd expect in this price range; the usual pica and elite fonts, and it accepts standard Epson codes for print variations, emphasized, double-strike and so on. There are no flashy self-diagnostic tests, international character sets and the like, but then no other printer under £250 is likely to have these features. The only omission worth mentioning is that you can't seem to underline text when in NLQ mode. This might seem like a bit of a drawback if you want to do a lot of heavy duty wordprocessing but I can quite happily live without it.

Once I figured out how to get the paper in it printed out text and screen dumps quite happily, and the print quality was perfectly acceptable. The NLQ printing is very good, though it is a little slow. This is the only area where the 2000's performance suffers in comparison with more expensive printers (though it is only in NLQ that the speed is the drawback), but if you've got a lot of text to print out there's no reason why you can't go off and have a cuppa while you're waiting. If you want more speed then you'll have to look at more expensive machines (and there is a DMP3000 and DMP4000 further up the range). I didn't feel that the slow NLQ printing was such a weakness that it made the 2000 unsuitable for wordprocessing. It might fall into the 'no-frills' category, but it's probably as good as you're likely to get for under £200.

CROSSWIRES

Ray Elder grapples with more readers' problems

QL Database

Q *Dear Mr. Elder,*
I wonder if you can help me, I'm a QL owner and I'm trying to write a database which is simpler than Archive to use.
I do all my programming in Basic, do not possess an assembler but I need a machine code routine that I can call from Basic.
The routine I need should enable random file access such as that one could say:— OPEN£4,dvn1_test: index£4,30:print£4,zxy$.
The point being that the QL internal file pointer is not available from Basic.
Admittedly it is reset to BOF when the file is opened an EOF can be tested for, but this is all on a record basis. I need a byte pointer such that I can read from the byte to the next CHR$ 10 separator or write from the current byte position with print£ in the usual way.
However setting the file byte pointer is only half the battle, I need a routine to ascertain where the current pointer on any open channel is currently situated. Using my example I would wish to say LET point=(index)£4:point= point+reclen:index£4,point:print£4, first_field_of_next_records$
This is a very tall order but I would be very pleased if you could help.
J.A. Powell, Staplehurst, Kent

A Phew! What you require is comprehensive random read/write access, and the answer is, unfortunately, that it cannot be done (as far as I know, I'd be happy to be proved wrong!).

Specific files can be read by keeping your own pointer, opening the channel making that number of reads and closing the file. This tends to be time consuming and of course you cannot write to a file.

The method most systems with non random read/write filing systems use is to read the complete file into an array(s) a the beginning of a session, perform all operations on the array(s) in Basic and then erase the old file and write the modified one back to microdrive.

With the QL's large memory this should cover a file of several thousand entries, be fast in operation and easy to manipulate with sorts etc. Some care and thought in the file structure will also maximise the number of entries, and the way in which they are stored.

I have written fileprograms for the Spectrum and TRL disk and for the BBC which has reasonable random access, and in all cases found it best to store each record as a single array$ entry, decoding each element or section into predefined fields.

I would recommend "The Working Sinclair QL" by David Lawrence and published by Sunshine as having a superb section on writing file programs.

A Cry For Help

Q *Dear Sir,*
I have a Spectrum 128+Opus Discovery Disk Drive with Centronics printer interface and a Tandy CGP115 Colour printer plotter.
My problem is the following:— apart from using the printer for listings it just sits there gathering dust. As you probably guessed I am not very good at programming.
What I need is a program(s) to help me print graphics, eg. screen strings and the like. If you know of any programs or can put me in touch with any readers who can help me it would be greately appreciated.
Yours Sincerely,
A. O'Neill, Coventry

A I assume that you are particularly interested in programs to print in colour, and as far as that goes I cannot help, I don't even know of anyone who has a colour printer!

As far as printer usage goes I use mine most of all for printing text and constantly use the Tasword wordprocessor. The second biggest use is for listings and lastly comes the occasional artwork.

The OCP Art Studio has many Centronics printer options and there should be one there that will work and this should enable screen$ to be printed — albeit in B/W.

Any reader who can help with colour printing information can contact Mr. O'Neill via ZX.

Tasword 3

Q *Dear Sirs,*
I am writing to you hoping that you'll be able to help me out with my Tasword 3 wordprocessor. This program works perfectly on my Spectrum connected to a Brother HR5 serial printer (RS232C) through Interface 1.
However when the same computer is connected to a Brother M1009 Centronics printer through the Tasman B interface, and although the software is customised exactly as prescribed in the Tasword manual, I cannot get the program to print the second character set or even the £ sign.
Am I doing something wrong or is the M1009 printer incompatible with the Tasword 3 program or is there something I've omitted?
Thanking you in advance,
Yours truly,
R.B. Michel, Bene, Switzerland.

A Well, the problem is that I don't know what you've tried. I suspect that you need to change the graphics codes to suit your printer. First consult the handbook for the codes which the printer recognises as character set change and also which character it uses for the sign, my sign is code 129.

Now go into define graphics and choose the code sequence that you are least likely to use and redefine it to the character change numbers — you may have to use another graphic to change it back again — then use another graphic set up to the pound sign code.

Now when printing from Tasword use the graphics you defined in the text or at the start of it to switch character sets. To print a use the graphic character and not the keyboard character for

The M1009 should operate perfectly, it is the same as the GLP printer and that functioned OK when we had one for review.

And A Friendly Word

... from our good friend and contributor John Wase on the topic of Tasword printer codes and justified text. Some time ago he wrote several detailed articles for us on just that topic and was moved to write after we published a letter in our Feb. issue.

Dear Ray,
Your correspondant in February's ZXC, R.F. Green of Rotterdam, has correctly noticed that Tasword's printer control codes destroy right justification.
However his solution, to incorporate code 32 in each control code sequence, is, at best, only partial.
In his solution control codes at the end or start of a line leave a space, moreover chemical formulae are impossible, or, for that matter, degrees centigrade.
The only real solution is to define double (32, 32) and triple (32, 32, 32) pads for a pair of control codes, and insert them appropriately and intelligently in the text.
Yours Sincerely
John Wase

QL COLUMN

Brian Becket on upgrade possibility for QL users

Digital Precision's Turbocharge Superbasic compiler has arrived at last. But alas it didn't arrive quite soon enough for me to play with for any length of time before sitting down to bash out this ever-readable column. So apologies for those waiting anxiously for further news about Turbo. So another month's wait for full details I'm afraid. Besides, I've already said so many nice thngs about the product in the last couple of months that you pretty much know what you need to know about it already. From my brief playings around, however, I can add that it looks to be as powerful and user-friendly as the company claims and, given that undue modesty isn't one of Digital's major faults, this is a strong recommendation in itself.

Sector Software informs me that its excellent multi-tasking, QL program **Taskmaster** (ZX Computing, February 1987) has been improved with lots of extra goodies. According to Sector, new Taskmaster features include: saved or re-drawn screens, specified re-draw keys sequences, flexible loading content and sequence, complete ICE compatibility, multiple copies from a "fantastic" (Sector's choice of words) code sharing facility, a "superb" notepad, eight specified programs from the entry screen, keydefine compatibility and constantly up-dated memory information of "To" and "From" devices. The "fantastic" code sharing facility is (in less dramatic no-PR English) certainly impressive. With 512K RAM to play with, Sector promises 12 "fully functioning" spreadsheets and 230K free memory.

The new Taskmaster is available on 3.5" disc as well as microdrive for those of you who can't be bothered to do the simple configuration necessary to transfer the microdrived program to disc. All copy protection (save the copyright itself) has apparently been removed so you can make as many masters as you need. Sector haven't sent me an up-dated Taskmaster (I could not in all honesty deny that there is a small element of a hint in those last few words) so I can only pass on the news but considering the quality of Taskmaster 1, I have little doubt that the new generation product makes an even more valuable addition to the QL software arsenal. If you already own Taskmaster, you can have the up-date for a fiver and a stamped and addressed Jiffy bag (the mind boggles) which is unquestionably reasonable. If you are considering buying it, it's only common sense to save the Fiver and go for the up-date the first time. But remember that you have to have at least 256K RAM to run Taskmaster and that an unexpanded QL won't do. Taskmaster (the last I heard) costs a reasonable £25 and (with enough RAM) will run all four Psion programs at the same time.

Driven to discs

If you are going to keep the old black box and stay with your QL rather than chuck it out, you are going to have to face up to the disc drive problem. If you haven't already invested in a disc drive, the time for action has arrived. Microdrives are becoming rare on the ground and should be an extinct species (many will argue that they should have been an extinct species from their very conception) in the not too distant future. As you have probably found out to your sorrow, the wretched things are undependable, temperamental, unpredictable and otherwise annoying to anybody save the most diehard masochist. Maybe I'm just unlucky but I've had endless numbers wear out or act up lately. My QL APL master and back-up have both given up the ghost and I'm fortunate to have it on disc as well. Just about everybody I know has had similar experiences. When blank microdrives were abundant, it wasn't too bad save for the two quid a go necessary to keep back-ups of the back-ups available at all times. But now that the microdrive is about to join the Dodo in the outer darkness, a disc drive is going to be the only way to keep the QL turning over regularly enough to justify the name of "micro-computer".

Several stalls were selling guaranteed, used microdrives at the last couple of Microfairs but, no criticism of the sellers, any working microdrive is an uncertain proposition be it used or new. If you're very careful, selective shopping for spare microdrives, judicious use of various cartridge doctors and media managers and the odd prayer will keep your QL ticking over on microdrives alone but it now seems silly not to put the same energy into getting a good disc drive at a reasonable price. Most of the new QL software (what there is) is either available on disc or can be transfered to disc without a hassle. Some of the older stuff takes an expert to get in and move it all over to a disc but — if you aren't up to the job — I think you will find the software houses mostly cooperative (although this co-operation might cost a couple of quid) in providing you with your favourite program on disc. If this doesn't work, you can devote your microdrive shopping time in the electronic antiques market and your microdrive first-aid and patching up time to those packages you can't get on disc but, I suspect, everything worth having will be made available for discs sooner or later.

There are any number of QL disc drives on the market and I don't propose to go over them now. What to do is to look around for bargains as they can be found and are likely to become more common as the QL fades out of the active market. In February, I mentioned that **ABC Elektronic** were preparing to release a QL up-grade kit costing a couple of hundred pounds and consisting of an IBM-style keyboard and a main-board box with built in 3.5" disc drives. The package will be marketed in this country by Digital Precision and, all being well, should be available about now. Had all been well, it should have been available a few weeks ago but I gather there was a delay of some six weeks on the West German end. Life being what it is, we must suppose that all won't be well for a bit longer but, if you are interested, keep checking with Digital. For those of you who haven't either yet invested in a disc drive or taken a hammer to the QL, this kit promises to be the answer. Not only do you get the disc facility but a handsome keyboard, a compact unit and a real on-off switch as well. Watch this space.

Multiface 128

John Wase gives the latest multi purpose interface a thorough testing.

Multiface 128
Romantic Robot
£44.95.

Yes, it's here at last. The all-singing, all-dancing Multiface 128 by Romantic Robot. Advertised as "No 1 Spectrum multipurpose interface", its principal purpose is to transfer tape-based software to other more rapid storage media — microdrives, Discovery, Beta or Disciple, or even to tape (with a turbo-load option). Although the copy saved is unprotected, it is of no use to commercial software pirates, as the first screen, usually the loader picture, is distorted to a greater or lesser extent in the top half. This is usually righted during the running of the program. As an alternative to mere copying, Multiface will allow you to stop a program, study it , with various displays of the registers in hex or decimal, even ASCII to show hidden text, PEEK at it, insert POKES and save screens.

The customary black plastic box has a through port at the bottom and a red spring-loaded button on top, and it fits upright at the back of the Spectrum, on the expansion port. I found it fitted well on a 48K, but not quite so well (though adequately) on the 128K Spectrum+2.

Snapshot

You load a program. Any old program. Press the red button; everything stops and a menu comes up on the bottom line inviting you to exit (to BASIC), return (to continue the program), save (to go to save menu), tool (to access multi-toolkit routines), print (to dump a screen to the printer, but only for interfaces with a COPY command like Kempston E or LPRINT III), jump (allows you to jump to any specified address on restart) and clear (to clear the extra 64K memory banks of a 128K Spectrum). I seized the nearest 128K program to hand; Tasword +2, loaded it and filled the file full of junk. I was particularly anxious to see if the distortion of the screen caused any problems; but it didn't. It saved a treat and loaded fine, the screen distortion disappeared.

The only point that I could mention was the loader screen being completely blue, it left Tasword +2 with a blue border until I did some operation which turned it green.

Transparency

Those who are familiar with the old Multiface 1 will know that it had a Kempston joystick and a switch to make the device transparent to other add-ons. There is no longer a joystick port, probably because the Plus2 has its own. The transparency switch is a different matter, however, and is sometimes necessary. This has been replaced by a software switch, toggled on and off by 'o' — it shows on the bottom line of the main menu along with the version number — in my case version number 87.1. Once set at off, Multiface is transparent on return to the program until reactivated with the red button.

I then tested it with some pictures. These saved perfectly, bottom lines and all. So you can save pictures of games and then print them out large on a printer to prove that you've got to that impossible screen, or you can even print out a poster advertising Aunt Jane's party.

Next I tried tape. First class. A battered old copy of Manic Miner saved perfectly but took well over 4 minutes to reload; the turbo version took 2 minutes 2 seconds, a very worthwhile saving.

Then I tried the Disciple interface. Things started to go wrong here. It just wouldn't. I tried it between the Spectrum and unit as recommended. Crash! Romantic Robot told me that a Disciple-compatible Multiface 128 would be in the shops in April. So take care.

Tool kit

And so to the other facilities. Like the toolkit. This menu gives 'q' (to quit), ENTER PEEKS and scrolls through addresses, and allows you to POKE any value into any address. The space bar allows you to type in a new address to POKE to, the current address being shown on the bottom line in either hex or decimal, being toggled between the two by 'h'. W is extremely powerful; it opens a window which displays 128 hex addresses and their contents, all in glorious black or yellow, the current address, the one on the bottom line, being also shown on the window in tasteful flashing blue. The whole can be scrolled around with the cursor keys. If, like me, your hex is doubtful, then it's easy to have the main window in hex and change the current address to decimal to check. This too changes as the window scrolls. The window can also be changed to ASCII equivalent, allowing "hidden" text to be located, and this can be a useful facility. Finally, 'reg' shows the Z80 registers at the time of stopping, and s selects additional RAM banks to examine. Incidentally, you need to POKE 8192/3 with a specific address, before returning to the main menu, if you want to use the jump facility. That, then is the address to which the program will subsequently jump.

Finally, if you're a machine code freak, there is an additional facility. Multiface has an 8K ROM and an 8K RAM as buffer, and this can be used for further machine code; it's no use trying BASIC, since the RAM overshadows the Spectrum ROM, so both cannot be paged in at the same time. For instance, it could be used for a piece of code enabling one to remove the menu from the bottom of the page thus pressing the button would freeze the frame and allow a photograph to be taken. The ideas and potential are limitless. . .

It's a pity about the temporary problem with the Disciple but, apart from that, Multiface 128 has a lot going for it. It has an excellent package of facilities, just the job for one to get the most out of ones gear. I thought it worked very well, and though it's not cheap, would recommend it.



HARDWARE

THE DISCOVERY COLUMN



More advice for Discovery owners, presented by John Wase.

I had vowed I would include no more cataloguing programs in the column. That was until I had this month's offerings. *Discovery Disc Map* from Tom Watts of Moenchengladbach, W. Germany, was so irresistable that I had to include it. Tom sent a disc, together with listings of all the program sections and voluminous notes which helped me very considerably.

This program has two main functions: firstly it prints out an *alphabetical* list of files (the software approach as Tom calls it). To do this it reads a catalogue and sorts it by machine code for speed; this uses a "shuttle" sort technique. The resulting printout also lists free areas on the disc. Secondly (the hardware approach) a *disc map* is produced. This shows the position of each file on the disc, together with those areas of the disc which are empty. This guides the user as to when to compact the disc.

Longcat

This routine has as a basis S. Nutting's "supercat" routine (ZX Computing, Oct '86); however, this allowed for only the standard size catalogue. In November '86 we printed a routine provided by Opus to extend the catalogue file to 676 entries; the other 43 blocks are then required for the catalogue file itself (43 + 676 = 677th file Discovery refuses to use). This increased catalogue size is often necessary when using a second drive of high capacity: I have had a lot of correspondence about adding 80T DSDD drives lately. Allowance for this increased size merely demands the use of different addresses; the necessary changes to S. Nutting's machine code routine are given in Figure 1 (the relevant lines in the BASIC program also need corresponding changes).

First, enter or load the machine code given in S. Nutting's routine from October ZX Computing (and reprinted in figure 1), and amend by POKEing the revised decimal values listed in figure 1a into the appropriate locations. Figure 1 also lists the changed instructions. Save this code with the command **SAVE *1;"catc"CODE 64900,100.** Next, enter the code in figure 2, using either an assembler, or the loader program (figure 3) which includes checksums to ensure correct entry: in this case, type the decimal numbers from the left hand side of figure 2 into the data statements, lines 100 onwards. (Do not, of course, type in the addresses 46000, etc.; these are automatically provided by the loader program). Finally, type in the BASIC program (figure 4) saving it with **SAVE *1;"ecat" LINE 10.** At line 760, between the two double quotes after TAB 27, type in the control codes for blue paper (extended 1) and for white ink (caps-shifted extended 7); between the next two double quotes type in the codes for white paper (extended 7) and for black ink (caps-shifted extended 0).

The Basic

Lines 10-60 define the functions used, initialise the variables and test for the presence of machine code, loading it as necessary. Lines 100-210 check on whether a print to screen, printer or both is needed, and whether you need a map by block, or an alphabetical file list. Lines 250-290 find the size of the catalogue file, POKEing it into the machine code, which then runs and reads in the file. Lines 300-510 set up headings, check the number of files on the disc and call the subroutine at line 1200 for a file list. Most of the original program went into lines 600-800, with additions for handling the screen/printer choice. The subroutine at lines 1000-1095 handles the end of disc printing, producing a list of all free areas for the alphabetical list, and in both cases a summary of used and free space, whilst the subroutine at lines 1100 to 1130 provides an appropriate line for the disc map. Finally, lines 1200-1250 contain a subroutine used only for alphabetic listings, which sets

Figure 1. Machine code loader.

```
   5 REM "TO ENTER CODE"
  10 CLEAR 64899: LET c=0: FOR a
=64900 TO 64999: READ n: POKE a,
n: LET c=c+n: NEXT a
  20 IF c<>11267  THEN  PRINT "D
ATA ERROR"
  30 STOP
  40 DATA 205,8,23,6,0,247,18,34
,226,253,33,0,0,17,48,242,14,0,2
05,218,253,20,69,62,6,184,40,3,3
5,24,241,33,46,249,221,33,50,242
,17,10,0,25,235,1,16,0,221,9,221
,229,213,221,78,2,221,70,3,33,25
5,255,237,66,40,16,221,110,0,221
,102
  50 DATA 1,14,7,205,218,253,225
,221,225,24,214,225,221,225,195,
72,23,197,213,229,62,1,6,2,205,0
,0,225,209,193,201


AMENDMENTS TO SUPERCAT ROUTINE

                 Location New value

                   64914     252
                   64915     208
                   64932     154
                   64933     181
                   64936     254
                   64937     208

The relevant instructions become:

                  LD    DE,53500
                  LD    HL,46490
                  LD    IX,53502
```

up all necessary details for the free space summary, POKEs the machine code with the number of files and runs the code to sort the list.

When you run the program, you are given the option of output to the screen, the printer or both. A sample output to the printer of the alphabetical list is given in figure 5, and of the disc map in figure 6; screen output, also of the disc map, is given in figure 7. Note how the original "supercat" program output is modified by the addition of the free space sections. The control codes to the printer are given in figure 8, together with the lines in which they appear. These worked well with my old Epson FX 80, but had to be modified for a daisywheel printer.

The Sort

And finally, a note on the sort: this is not strictly alphabetical as sorting is merely character by character using the ASCII codes. Thus a file called A10 will come before file A2 because 1 is less than 2; similarly FILEZ will come before FILEa because upper case letters have lower ASCII codes than lower case. It would be possible to compensate for these variations, but only at the expense of speed and large amounts of code, and this just isn't worth while.

Help

A number of the letters I have recently received comment on previous items or ask for help. Thus Peter Marks, of Stirchley in Birmingham writes about multifaces, mentioning that there are two versions of the 48K multiface 1, and that whilst the older one is incompatible with the verion of 2.2 Discovery ROM, the new one works, but only when placed in betweeen Discovery and Spectrum, thereby fouling the disc slots. He found he had to use twinstick, too, to stop things wobbling around. Well, Peter, it won't work on the output port of Discovery, and neither will the Disciple snapshot button for that matter, because they both use the non-maskable interrupts (NMI) line, and this is the one line which is not continued through to the Discovery output port, number 13 on top, from the keyway end of the Spectrum port (see Spectrum manual). If you want to void your guarantee, solder an insulated wire across from the Spectrum side to the output port of the Discovery circuit board, and multiface will sit there tightly instead of fouling the disc inlet.

And the customary printing problems . . .

Figure 2.

```
     Sort routine for disc map utility.

                               ORG  46000
SRT   46000 033,028,209        LD   HL,53532    ;address of 2nd file data
      46003 221,033,174,181 LD   IX,46510    ;address of 2nd file header
      46007 237,075,128,180 LD   BC,(FILES) ;number of files on disc
LOOP  46011 011                DEC  BC
      46012 120                LD   A,B         ;test if zero
      46013 177                OR   C
      46014 200                RET  Z           ;all files checked
      46015 017,112,180        LD   DE,STORE    ;address of storage area
      46018 197                PUSH BC          ;preserve
      46019 001,016,000        LD   BC,16
      46022 237,176            LDIR             ;store file data
      46024 229                PUSH HL          ;hold address reached
      46025 175                XOR  A           ;clear C flag
      46026 001,026,000        LD   BC,26       ;point to previous file name
      46029 237,066            SBC  HL,BC
      46031 205,223,179        CALL TEST        ;test names
      46034 048,002            JR   NC,L1       ;C set and Z clear means
      46036 032,023            JR   NZ,MOVE     ;stored data out of sequence
L1    46038 001,010,000        LD   BC,10       ;increment header pointer
      46041 221,009            ADD  IX,BC
      46043 225                POP  HL          ;recover data pointer
      46044 193                POP  BC          ;recover count
      46045 024,220            JR   LOOP        ;test next file
TEST  46047 017,118,180        LD   DE,(STORE+6) ;stored file name
      46050 006,010            LD   B,10        ;chars in name
T1    46052 026                LD   A,(DE)      ;next char
      46053 190                CP   (HL)        ;compare with (HL)
      46054 216                RET  C           ;inequality: (HL) > store
      46055 192                RET  NZ          ;inequality: store > (HL)
      46056 019                INC  DE          ;next char
      46057 035                INC  HL
      46058 016,248            DJNZ T1
      46060 201                RET              ;equal: retn C clear Z set
MOVE  46061 209                POP  DE          ;file reached
      46062 193                POP  BC          ;files left
      46063 197                PUSH BC
      46064 213                PUSH DE
      46065 042,128,180        LD   HL,(FILES) ;total files
      46068 175                XOR  A           ;clear C flag
      46069 237,066            SBC  HL,BC       ;files checked
      46071 043                DEC  HL          ;files to test
      46072 235                EX   DE,HL
      46073 001,042,000        LD   BC,42       ;get to current file - 1
      46076 237,066            SBC  HL,BC
      46078 001,016,000        LD   BC,16       ;set up bytes to move
      46081 237,067,130,180 LD   (BYT1),BC   ;file data
      46085 001,010,000        LD   BC,10
      46088 237,067,132,180 LD   (BYT2),BC   ;file headers
M1    46092 122                LD   A,D         ;test end
      46093 179                OR   E
      46094 040,037            JR   Z,M2        ;end reached
      46096 213                PUSH DE          ;preserve
      46097 229                PUSH HL
      46098 205,223,179        CALL TEST        ;test for right slot
      46101 225                POP  HL          ;recover
      46102 209                POP  DE
      46103 048,028            JR   NC,M2       ;slot found
      46105 027                DEC  DE          ;decrement count
      46106 001,016,000        LD   BC,16       ;adjust address
      46109 175                XOR  A           ;clear C flag
      46110 237,066            SBC  HL,BC
      46112 229                PUSH HL          ;adjust byte totals
      46113 042,130,180        LD   HL,(BYT1)
      46116 009                ADD  HL,BC
      46117 034,130,180        LD   (BYT1),HL
      46120 042,132,180        LD   HL,(BYT2)
      46123 001,010,000        LD   BC,10
      46126 009                ADD  HL,BC
      46127 034,132,180        LD   (BYT2),HL
      46130 225                POP  HL
      46131 024,215            JR   M1          ;next file name
M2    46133 209                POP  DE          ;next file
      46134 213                PUSH DE
      46135 033,240,255        LD   HL,-16
      46138 027                DEC  DE
      46139 025                ADD  HL,DE       ;one file down
      46140 237,075,130,180 LD   BC,(BYT1)   ;bytes to move
      46144 237,184            LDDR
      46146 035                INC  HL          ;address for stored data
      46147 017,112,180        LD   DE,STORE
      46150 235                EX   DE,HL
      46151 001,016,000        LD   BC,16
      46154 237,176            LDIR
      46156 221,229            PUSH IX          ;address of header
      46158 225                POP  HL
      46159 017,112,180        LD   DE,STORE
      46162 001,010,000        LD   BC,10
      46165 237,176            LDIR             ;header to store
      46167 043                DEC  HL          ;address for header data
      46168 235                EX   DE,HL
```

Mr Davies of Hanworth writes that he has not been able to get Tasprint or Tascopy to work with a Spectrum 48K, Saga 1, Opus Discovery and Citizen 120D printer. And D.R.G. Shepherd of Lenzie is having difficulty in getting Omnicalc 2 and OCP's Finance Manager to print using a 48K Spectrum, Opus Discovery and Brother M1009 printer. Can anyone out there help, please?

Code Machine

These printer problems are slowly being solved by our readers. Here is a short routine by T. Collier of Edinburgh, which allows all printger output from the *Code Machine* by Picturesque to be printed on an 80 column printer through the Opus printer port.

(1). Type in and run the loader program (figure 9). Two versions of the code will be saved on tape, one for the assembler and one for the monitor.

(2). Load the Assembler. When PRINTER OPTIONS are displayed, select Key 4 (DK'Tronics/Centronics). Load the "ASSDK" CODE saved by the loader program. When the prompt for LINE FEED and CARRIAGE RETURN is displayed, stop the tape and press symbol shift/Key A, then **ENTER.** Type in LIST 600 and break; insert **CLOSE # 3;"b" :** after the line number, re-enter the revised line 600 and then enter as a direct command **OPEN # 3;"b" : GO TO 300.** The SAVE options should be displayed on screen. Insert a formatted disc and press key 2 for a microdrive save.

(3). Load the Monitor instead of the Assembler, following the instructions in section 2, but loading "MONDK"CODE.

An assembly listing of the printer routine is given in figure 10.

Labels

All right, I asked for it. I have been inundated with random access filing programs. Here is one that is just a little out of the ordinary, by Richard Lown of Wivenhoe. And it's really one to get your teeth into. It demonstrates three things. Firstly, it shows how a fairly complex random access disc filing system can work. Secondly, it is written in Beta Basic. For the few of you who don't know this program, let me say that it adds a great many commands to standard Sinclair Basic, including procedures. It allows you to save procedures separately, so that a library of these can be built up:

```
      46169 033,246,255      LD   HL,-10
      46172 025              ADD  HL,DE        ;data to be moved
      46173 237,075,132,180  LD   BC,(BYT2)    ;bytes to be moved
      46177 237,184          LDDR
      46179 035              INC  HL           ;move in stored header
      46180 017,112,180      LD   DE,STORE
      46183 235              EX   DE,HL
      46184 001,010,000      LD   BC,10
      46187 237,176          LDIR
      46189 195,214,179      JP   L1           ;return to main loop
STORE 46192 000,000,000,000  DEFS 16
            000,000,000,000
            000,000,000,000
            000,000,000,000
FILES 46208 001,000          DEFW 1
BYT1  46210 000,000          DEFW 0
BYT2  46212 000,000          DEFW 0
```

Figure 3. Loader program.

```
 10 LET s=0
 20 FOR i=46000 TO 46099: READ a: POKE i,a: LET s=s+a: NEXT i
 30 IF s<>11536 THEN PRINT "ERROR IN DATA LINES 100 - 109": STOP
 40 FOR i=46100 TO 46213: READ a: POKE i,a: LET s=s+a: NEXT i
 50 IF s<>22801 THEN PRINT "ERROR IN DATA LINES 110 - 121": STOP
100 DATA 33,28,209,221,33,174,181,237,75,128
101 DATA 180,11
.............(Data from listing 2)
```

Figure 4.

```
 10 DEF FN a(x)=PEEK (x+a)+256*PEEK (a+x+1)
 20 DEF FN b()=CODE INKEY$#0+256*CODE INKEY$#0
 30 IF PEEK 64900<>205 AND PEEK 64901<>8 THEN CLEAR 45999: LOAD *1;"catc"CODE :
LOAD *1;"srt"CODE
 40 CLS #
 50 DIM e$(678,3): DIM p$(11): DIM q$(16): DIM r$(7): DIM s$(7): DIM t$(7): DIM
u$(7): DIM v$(8): DIM w$(6): DIM x$(4): DIM c$(5): DIM d$(7)
 60 LET sub=1: LET bl=0: LET p=0: LET p1=0: LET scr=1
100 PRINT AT 5,8; INVERSE 1;"?? PRINTER ON ??"
110 IF INKEY$<>"" THEN GO TO 110
120 LET z$=INKEY$: IF z$="n" OR z$="N" THEN GO TO 150
130 IF z$<>"y" AND z$<>"Y" THEN GO TO 120
140 LET p=1: OPEN #3;"b": LPRINT
150 PRINT AT 7,1; PAPER 1; INK 7;"?? MAP (M) OR FILE LIST (L) ??"
160 LET z$=INKEY$: IF z$="l" OR z$="L" THEN LET p1=1: GO TO 180
170 IF z$<>"m" AND z$<>"M" THEN GO TO 160
180 IF NOT p THEN GO TO 250
190 PRINT AT 9,7; PAPER 3; INK 7;"?? PRINTER ONLY ??"
200 LET z$=INKEY$: IF z$="y" OR z$="Y" THEN LET scr=0: GO TO 250
210 IF z$<>"n" AND z$<>"N" THEN GO TO 200
250 INPUT "Drive: ";d: POKE 64990,d
260 CLEAR #0: OPEN #0;" CAT ";dRND16:POINT #0;1
270 LET c1=FN b()*FN b()*0+FN b()
280 CLEAR #0: POKE 64924,c1
290 RANDOMIZE USR 64900
300 CLS #
310 PRINT "TITLE = ";: FOR a=53506 TO 53515: PRINT CHR$ PEEK a;: NEXT a: PRINT
" ";
320 IF NOT p THEN GO TO 370
330 LPRINT CHR$ 14;CHR$ 27;"-1CARTRIDGE NAME = ";: FOR a=53506 TO 53515: LPRINT
 CHR$ PEEK a;: NEXT a: LPRINT CHR$ 27;"-0";CHR$ 10;CHR$ 10
340 LPRINT "  NAME            TYPE         BYTES START   END  BLOCKS PROG/ADDR VARS
  AUTO"
350 LPRINT "  ____            ____         _____ _____   ___  ______ _________ ----
  ____";CHR$ 10;CHR$ 10
360 LET p$="CAT": LET q$="CATALOGUE": LET a=53500: LET e=FN a(4): LET s$=STR$ F
N a(2): LET t$=STR$ e: LET u$=STR$ (e-VAL s$+1)
370 LET i=0: FOR a=53516 TO 53500+16*677 STEP 16: IF FN a(4)<>65535 THEN LET i=
i+1: NEXT a
380 PRINT "(";STR$ i;" FILES)": IF NOT p THEN GO TO 500
390 LET v$="FILES = ": LET w$=STR$ i
400 LPRINT p$;q$;r$;s$;t$;u$;v$;w$;CHR$ 10;CHR$ 10
500 IF p1 THEN GO SUB 1200
510 LET a=53516: LET i=46500: LET e1=c1
600 LET b=FN a(0)
610 LET s=FN a(2)
620 LET e=FN a(4)
630 LET mem=(e-s)*256+b-6
640 IF e=65535 THEN GO SUB 1000: STOP
650 IF NOT p1 THEN GO SUB 1100
660 FOR k=a+6 TO a+15: IF scr THEN PRINT PAPER 4;CHR$ PEEK k;
670 LET p$(k-a-5)=CHR$ PEEK k: NEXT k
680 IF scr THEN PRINT mem;TAB 17;s;TAB 21;e;TAB 25;e-s+1
690 LET r$=STR$ mem: LET s$=STR$ s: LET t$=STR$ e: LET u$=STR$ (e-s+1)
700 LET t=a: LET a=i: LET type=PEEK a
710 LET length=FN a(1)
720 LET start=FN a(3): IF FN a(3)>32767 AND type=0 THEN LET start=0
730 LET basic=FN a(5)
740 IF type>3 THEN LET type=4
750 GO TO 760+20*type
760 IF scr THEN PRINT PAPER 1; INK 7;"Basic+Variables Basic Vari Auto BASIC:";T
AB 10;length;TAB 16;basic;TAB 22;length-basic.;TAB 27;""+STR$ start AND start<>0;
"";" " AND start=0
770 LET q$="PROGRAM": LET v$=STR$ basic: LET w$=STR$ (length-basic): LET x$=STR
$ start: GO TO 860
780 LET n=PEEK (i+4)-64: IF scr THEN PRINT PAPER 3; INK 6;"NUMERIC ARRAY   ";me
```

▶

DISCOVERY

the procedures are called merely by calling their names in the main program. It is much more powerful and much less clumsy than using GO TO and GO SUB, forming the basis of much more structured and intelligible programming. Even if you don't know all the commands, have a look at it, I am sure it will intrigue you. Incidentally, readers might wish to know that a new, 128K version of Beta Basic, making extensive use of tghe RAM disc facilities will be out shortly. Finally, this present program illustrates the power of good BASIC and it actually prints labels with selected names and addresses, too.

Load Beta Basic, type in and save FILCRE and LABELS listings (figures 11 and 12). FILCRE should then be loaded first, as it creates files. One of the snags I found is that Richard obviously did everything with capslock down, and his program doesn't respond to lower case input: beware. FILCRE creates a file called M on the disc, so don't write-protect it. Then load LABELS. Line 13 starts the procedure OS_num which redefines the character set and all the numerals, making them a rather pleasant old style. The section from lines 30 to 85 loads the datafile disc (the file M which you've just made or got out of the box), and the main program consists of a big DO loop from lines 90 to 120, with the practical procedures listed in line 110. Proc tel searches for a telephone number (lines 500-600). Proc add (lines 800-900)

```
m;TAB 21;" TYPE=";CHR$ n
 790 LET q$="NUMERIC ARRAY": LET v$=CHR$ n: LET w$="": LET x$="": GO TO 860
 800 LET c=PEEK (i+4)-128: IF scr THEN PRINT PAPER 3; INK 6;"CHARACTER ARRAY ";m
em;TAB 21;" TYPE=";CHR$ c;"$"
 810 LET q$="CHARACTER ARRAY": LET v$=CHR$ c+"$": LET w$="": LET x$="": GO TO 86
0
 820 IF scr THEN PRINT PAPER 6; INK 2;"MACHINECODE ";start;",";length
 830 LET q$="MACHINE CODE": LET v$=STR$ start: LET w$="": LET x$="": GO TO 860
 840 IF scr THEN PRINT PAPER 5;"PRINTFILE TYPE  LENGTH=";mem
 850 LET q$="DATA FILE": LET v$="": LET w$="": LET x$=""
 860 IF x$="0    " THEN LET x$=""
 870 IF p THEN LPRINT p$;q$;r$;s$;t$;u$;v$;w$;x$;CHR$ 10;CHR$ 10
 880 LET el=e: LET a=t: LET a=a+16: LET i=i+10: GO TO 600
1000 LET s=s+1: DIM c$(5): IF NOT p1 THEN GO SUB 1100: GO TO 1040
1010 IF p THEN LPRINT CHR$ 10;CHR$ 27;"-1FREE AREAS ON DISC:";CHR$ 27;"-0";CHR$
10;CHR$ 10
1020 IF p THEN LPRINT "FROM TO BLOCKS BYTES      FROM TO BLOCKS BYTES      FROM TO
 BLOCKS BYTES";CHR$ 13;"____ __ ______ _____      ____ __ ______ _____      ____ _
_ ______ _____";CHR$ 10;CHR$ 10
1030 IF p THEN FOR a=1 TO sub STEP 6: GO SUB 1090: LPRINT CHR$ 10: NEXT a
1040 IF p THEN LPRINT CHR$ 10;"BLOCKS ON DISC = ";s;"  (";256*s;" BYTES)";CHR$ 1
3;CHR$ 10;"USED: ";s-bl;" (";256*(s-bl);" BYTES)";CHR$ 13;CHR$ 10;"FREE: ";bl;"
(";256*bl;" BYTES)"
1050 IF scr THEN PRINT "FREE:  ";bl;" Blocks (";256*bl;" Bytes)"
1060 IF p THEN CLOSE #3
1070 RETURN
1090 FOR k=0 TO 4 STEP 2: LET c$=e$(a+k): LPRINT c$;: LET c$=e$(a+k+1): LPRINT c
$;: IF c$<>"     " THEN LET c$=STR$ (VAL e$(a+k+1)-VAL e$(a+k)+1): LPRINT c$;: L
ET c$=STR$ (256*VAL c$): LPRINT c$;"     ";
1095 NEXT k: RETURN
1100 IF el+1>=s THEN RETURN
1105 LET bl=bl+s-el-1: LET d$=STR$ (256*(s-el-1))
1110 IF p THEN LPRINT "              UNALLOCATED      ";d$;el+1;(" " AND el+1<10);("
 " AND el+1<100);"    ";s-1;(" " AND s-1<10);(" " AND s-1<100);"    ";s-el-1;CHR
$ 10;CHR$ 10
1120 IF scr THEN PRINT INK 7; PAPER 0;"FREE SPACE"; INK 0; PAPER 7;"   ";el+1;"
to ";s-1
1130 RETURN
1200 LET bl=0: LET el=c1: FOR a=53516 TO 53500+16*i STEP 16
1210 LET e=FN a(4): LET s=FN a(2): IF el+1<s THEN LET e$(sub)=STR$ (el+1): LET e
$(sub+1)=STR$ (s-1): LET sub=sub+2: LET bl=bl+s-el-1
1220 LET el=e: NEXT a
1230 LET s=FN a(2): IF e<s THEN LET e$(sub)=STR$ (e+1): LET e$(sub+1)=STR$ s: LE
T bl=bl+s-e
1240 POKE 46209,INT (i/256): POKE 46208,INT (256*(i/256-PEEK 46209)+.1)
1250 RANDOMIZE USR 46000: RETURN
```

Figure 5. Sample alphabetical list.

CARTRIDGE NAME = work1

NAME	TYPE	BYTES	START	END	BLOCKS	PROG/ADDR	VARS	AUTO
CAT	CATALOGUE		0	6	7	FILES = 10		
arrsample	NUMERIC ARRAY	53	109	109	1	A		
catc	MACHINE CODE	100	7	7	1	64900		
datatest	DATA FILE	3	108	108	1			
ecat	PROGRAM	4939	110	129	20	4939	0	10
srt	MACHINE CODE	214	10	10	1	46000		
srt-ltr	MACHINE CODE	1433	210	215	6	26502		
srt-sce	MACHINE CODE	5662	11	33	23	26502		
srt-txt	MACHINE CODE	6223	185	209	25	26502		
srtloader	PROGRAM	371	8	9	2	346	25	
strsample	CHARACTER ARRAY	2039	79	86	8	E$		

FREE AREAS ON DISC:

FROM	TO	BLOCKS	BYTES	FROM	TO	BLOCKS	BYTES	FROM	TO	BLOCKS	BYTES
34	78	45	11520	87	107	21	5376	130	184	55	14080
216	719	504	128902								

BLOCKS ON DISC = 720 (184320 BYTES)
USED: 95 (24320 BYTES)
FREE: 625 (160000 BYTES)

Figure 6. Sample disc map.

CARTRIDGE NAME = work1

NAME	TYPE	BYTES	START	END	BLOCKS	PROG/ADDR	VARS	AUTO
CAT	CATALOGUE		0	6	7	FILES = 10		
catc	MACHINE CODE	100	7	7	1	64900		
srtloader	PROGRAM	371	8	9	2	346	25	
srt	MACHINE CODE	214	10	10	1	46000		
srt-sce	MACHINE CODE	5662	11	33	23	26502		
	UNALLOCATED	11520	34	78	45			
strsample	CHARACTER ARRAY	2039	79	86	8	E$		
	UNALLOCATED	5376	87	107	21			
datatest	DATA FILE	3	108	108	1			
arrsample	NUMERIC ARRAY	53	109	109	1	A		
ecat	PROGRAM	4939	110	129	20	4939	0	10
	UNALLOCATED	14080	130	184	55			
srt-txt	MACHINE CODE	6223	185	209	25	26502		
srt-ltr	MACHINE CODE	1433	210	215	6	26502		
	UNALLOCATED	129024	216	719	504			

BLOCKS ON DISC = 720 (184320 BYTES)
USED: 95 (24320 BYTES)
FREE: 625 (160000 BYTES)

Figure 7. Screen output of disc map.

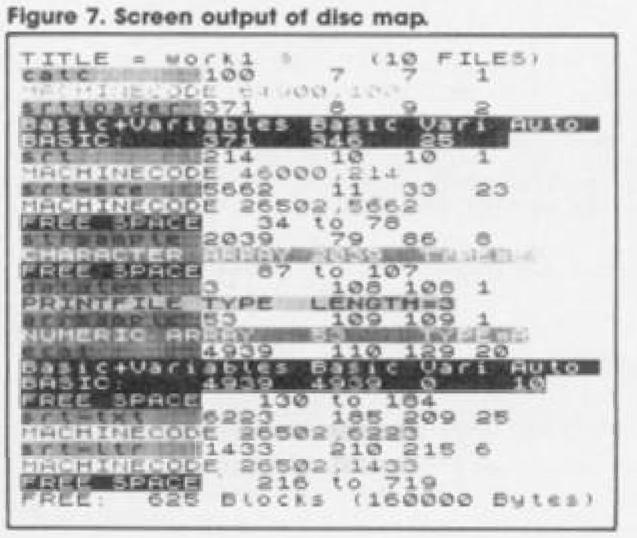


Figure 8.

PRINTER CONTROL CODES USED

Code	Meaning	Lines involved
CHR$ 14	Expanded print	330
CHR$ 27;"-1"	Underline	330,1010
CHR$ 27;"-0"	Underline off	330,1010
CHR$ 10	Linefeed	Most lines with LPRINT
CHR$ 13	Carriage return	1020 and implicit in most LPRINTs

adds names, addresses and telephone numbers to the lists, displays them for checking and saves them on the file: proc change (lines 1000-1200) similarly allows you to change files. Proc find (lines 2000-2100) is a string search routine. Proc display (lines 5000-5050) merely displays information, and proc exit, NEWs the program after displaying "FAREWELL" on the screen. A PAUSE would help here. Finally proc list lists files to the printer or screen. The program works rather nicely and would do very well for a club to keep a list of members which could be updated readily, and which would provide a list of lables for sending out, for instance, club magazines.

Finally . . .

Thanks for all your letters and programs. I hope that this column provides useful information even for non-disc users who might be curious, and will stimulate you into submitting further routines. Please keep them coming. See you next month.

Figure 9. T. Collier's loader program.

```
   1 REM *********************
   2 REM ***LOADER PROGRAM***
   3 REM ***  T. COLLIER  ***
   4 REM *** CODE MACHINE ***
   5 REM ***PRINTER OUTPUT***
   6 REM *********************
  10 CLEAR 31999: RESTORE 100: F
OR f=32000 TO 32041: READ a: POK
E f,a: NEXT f
  20 PRINT "Insert blank tape &
press a key": PAUSE 0
  30 SAVE "ASSDK"CODE 3200,42: P
OKE 32002,113: POKE 32003,102: P
OKE 32021,113: POKE 32022,230: S
AVE "MONDK"CODE 32000,42
 100 DATA 253,203,115,118,32,13,
245,229,213,197,62,3,205,1,22
 110 DATA 193,209,225,241,253,20
3,115,246,245,215,205,84,31,56,9
 120 DATA 241,245,13,55,200,62,1
3,24,240,241,167,201
```

Figure 10.

```
PRINTER ROUTINE FOR PICTURESQUE
ASSEMBLER/MONITOR & OPUS DISC DRIVE.

7D00                0000          ORG  32000
7D00 FDCB7376       0005          BIT  6,(IY+73H)
7D04 200D           0010          JR   NZ,CHOPEN
7D06 F5             0015          PUSH AF
7D07 E5             0020          PUSH HL
7D08 D5             0025          PUSH DE
7D09 C5             0030          PUSH BC
7D0A 3E03           0035          LD   A,3
7D0C CD0116         0040          CALL 1601H
7D0F C1             0045          POP  BC
7D10 D1             0050          POP  DE
7D11 E1             0055          POP  HL
7D12 F1             0060          POP  AF
7D13 FDCB73F6       0065 CHOPEN   SET  6,(IY+73H)
7D17 F5             0070 SENDCR   PUSH AF
7D18 D7             0075          RST  10H
7D19 CD541F         0080          CALL 1F54H
7D1C 3809           0085          JR   C,NOBRK
7D1E F1             0090          POP  AF
7D1F FE0D           0095          CP   0DH
7D21 37             0100          SCF
7D22 C8             0105          RET  Z
7D23 3E0D           0110          LD   A,0DH
7D25 10F0           0115          JR   SENDCR
7D27 F1             0120 NOBRK    POP  AF
7D28 A7             0125          AND  A
7D29 C9             0130          RET
                    0135          END
```

Figure 11.

```
  90 REM ***PROGRAM FILCRE***
  91 REM *** RICHARD LOWN ***
  99 REM FILE CREATION
 100 LET F$="LABELS": DIM M(1):
     LET M(1)=1
 110 add M(1),F$
 800 REM add M(1),F$
 810 DEF PROC add REF M(),F$
 820 CLS
 825 LOCAL Y$,Z,W
 826 DIM A$(208)
 830 DO
 840 INPUT "FIRST NAME ";A$(1 TO
     26): PRINT AT 0,0;A$(1 TO
     26): INPUT "SECOND NAME ";A
     $(27 TO 52): PRINT AT 1,0;A
     $(27 TO 52)
 844 LET Z=27: LET W=0
 845 DO UNTIL Z>130
 850 LET Z=Z+26: LET W=W+1: PRIN
     T AT 20,0;"N/L TO EXIT THIS
     PART"
 860 PRINT AT 21,0;"ADDRESS ";W
 870 INPUT A$(Z TO Z+25)
 880 PRINT AT W+2,0;A$(Z TO Z+25
     )
 890 EXIT IF A$(Z)=CHR$ 13
 900 LOOP
 910 PRINT AT 20,0;"

     "
 920 INPUT "POSTCODE ";A$(157 TO
     182): PRINT AT 11,0;A$(157
     TO 182): INPUT "TELEPHONE N
     O: ";A$(183 TO 208): PRINT
     AT 13,0;A$(183 TO 208): PRI
     NT AT 15,0;"IF THIS IS OK P
     RESS 'Y' ELSE 'N'"
 930 GET Y$: CLS
 940 LOOP UNTIL Y$="Y"
 942 SAVE "M" DATA M()
 945 OPEN #4;"M";1;F$RND209,-1
 948 POINT #4;M(1)
 950 PRINT #4;A$
 955 CLOSE #4
 960 END PROC
1000 STOP
```

Figure 12

```
   8 REM ***PROGRAM LABELS***
   9 REM *** RICHARD LOWN ***
  10 CLS : PRINT AT 10,5;"HOLD O
     N A MINUTE"
  13 OS_num
  14 CLS
  20 DEF PROC OS_num: LOCAL s,i,
     c: CLEAR 384: CLEAR 384: LE
     T s=DPEEK(23730)+2: DO UNTI
     L PEEK (s-1)=0: LET s=s+3+D
     PEEK(s): LOOP : POKE s,MEMO
     RY$()(15616 TO 16383): DPOK
     E 23606,s-256
  21 RESTORE 22: FOR i=s+128 TO
     s+207: READ c: POKE i,c: NE
     XT i: RESTORE
  22 DATA 0,0,56,76,84,100,56,0,
     0,0,48,16,16,16,56,0,0,0,56
     ,68,24,32,124,0,0,0,56,68,8
     ,4,68,56,0,0,8,24,40,72,252
     ,8,0,0,120,64,120,4,68,56,0
     ,16,32,120,68,68,56,0,0,0,1
     24,4,8,16,32,64,0,56,68,56,
     68,68,56,0,0,0,56,68,68,56,
     8,16
  23 FOR i=s+8 TO s+759: POKE i,
     OR(PEEK i,PEEK i/2): NEXT i
  24 END PROC
  30 PRINT "DATAFILE DISC? PRESS
     'Y':": GET Z$
  40 LET F$="LABELS"
  80 REM menu
  84 DIM M(1)
  85 LOAD *"m";1;"M" DATA M(): R
     ANDOMIZE USR 58419: CLS
  90 DO
  95 CLS
 100 PRINT AT 0,10;"LABELS ";AT
     2,0;;"1    OUTPUT";AT 4,0;"
     2 CHANGE RECORD";AT 6,0;"3
     ADD NEW ITEM.";AT 8,0;"4 LI
     ST   NAMES.";AT 10,0;"5
     EXIT"
 110 GET A: ON A:tel M(1),F$:cha
     nge M(1),F$:add M(1),F$:lst
     M(1),F$:exit F$
 120 LOOP
 500 DEF PROC tel REF M(),F$
 510 CLS
 520 PRINT AT 9,0;"IF YOU KNOW T
     HE NUMBER OF THE   ADDRESSE
     E THEN TYPE IT IN NOW,  OTH
     ERWISE PRESS 'G'"
 530 INPUT Y$
 540 IF Y$<>"G" THEN DIM S$(208)
     : OPEN #4;"M";1;F$RND209:PO
     INT #4;VAL Y$: INPUT #4;S$:
     CLOSE #4
 550 IF Y$="G" THEN CLS : LET PT
     R=1: INPUT "TYPE IN THE NAM
     E OF ADDRESSEE: "; LINE S$:
     find S$,M(1),F$,PTR
 560 IF Y$="N" THEN PRINT AT 19,
     0;"NOT FOUND:PRESS ANY KEY
     FOR MENU": PAUSE 0: END PRO
     C
 570 CLS
 580 PRINT AT 9,0;"PRESS P FOR P
     RINTER,S FOR SCREEN"
 590 DO
 600 GET Y$: IF Y$="P" THEN LET
     K=3: ELSE LET K=2
 610 LOOP UNTIL Y$="P" OR Y$="S"
 620 DO
 630 CLS : PRINT AT 2,0;"T TEL N
     O",,"A FULL ADDRESS",,"L LA
     BEL"
 640 GET D$
 650 LOOP UNTIL D$="T" OR D$="A"
     OR D$="L"
 660 CLS : IF D$="T" THEN PRINT
     #K;S$(1 TO 26): PRINT #K;S$
     (27 TO 52): PRINT #K;S$(183
     TO 208): INPUT "PRESS ENTER
     FOR MENU."; LINE Y$: END P
     ROC
 670 IF D$="A" THEN display# K,S
     $,8: INPUT "PRESS ENTER FOR
     MENU."; LINE Y$: END PROC
 680 IF D$="L" THEN RESTORE 5500
     : DO :display# 3,S$,7: PRIN
     T AT 11,2;"IF YOU WANT TO Q
     UIT PRESS 'Q'": GET Y$: LOO
     P WHILE Y$<>"Q"
 690 END PROC
 800 REM add
 810 DEF PROC add REF M(),F$
 820 CLS
 825 LOCAL Y$,Z,W
 826 DIM A$(208)
 830 DO
 840 INPUT "FIRST NAME ";A$(1 TO
     26): PRINT AT 0,0;A$(1 TO
     26): INPUT "SECOND NAME ";A
     $(27 TO 52): PRINT AT 1,0;A
     $(27 TO 52)
 844 LET Z=27: LET W=0
 845 DO UNTIL Z>130
 850 LET Z=Z+26: LET W=W+1: PRIN
     T AT 20,0;"N/L TO EXIT THIS
     PART"
 860 PRINT AT 21,0;"ADDRESS ";W
 870 INPUT A$(Z TO Z+25)
 880 PRINT AT W+2,0;A$(Z TO Z+25
     )
 890 EXIT IF A$(Z)=CHR$ 13
 900 LOOP
 910 PRINT AT 20,0;"

     "
 920 INPUT "POSTCODE ";A$(157 TO
     182): PRINT AT 11,0;A$(157
     TO 182): INPUT "TELEPHONE N
     O: ";A$(183 TO 208): PRINT
     AT 13,0;A$(183 TO 208): PRI
     NT AT 15,0;"IF THIS IS OK P
     RESS 'Y' ELSE 'N'"
 930 GET Y$: CLS
 940 LOOP UNTIL Y$="Y"
 942 LET M(1)=M(1)+1: SAVE "M" D
     ATA M()
```

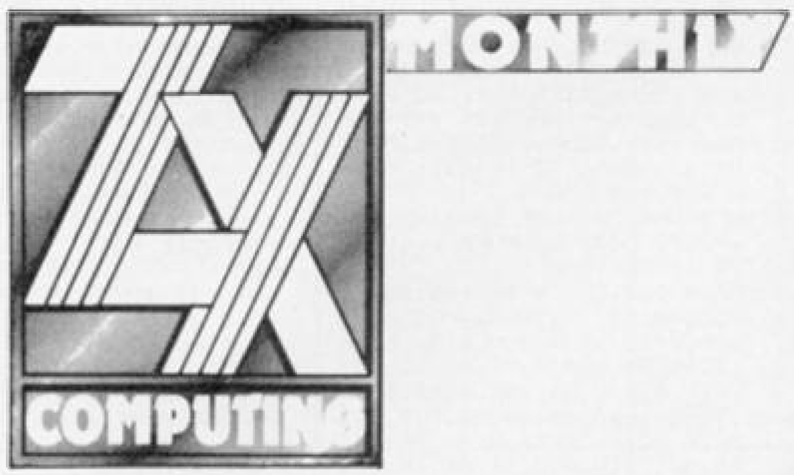
▶

DISCOVERY

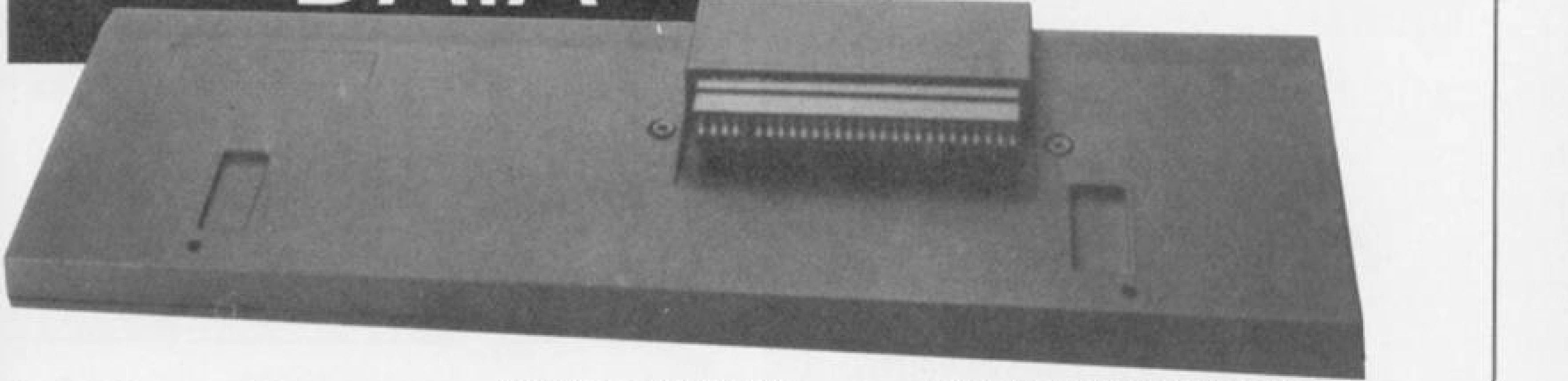
```
 945 OPEN #4;"M";1;F$RND209
 948 POINT #4;M(1)
 950 PRINT #4;A$
 955 CLOSE #4
 960 SAVE *"M";3;"M" DATA M()
 965 OPEN #4;"M";3;F$RND209
 970 POINT #4;M(1)
 975 PRINT #4;A$
 980 CLOSE #4
 999 END PROC
1000 DEF PROC change REF M(),F$
1004 LOCAL L,S,Y$,N,S$,PTR
1005 LET PTR=1
1006 CLS : PRINT AT 9,0;"IF YOU
     KNOW THE NUMBER OF THE   AD
     DRESSEE THEN TYPE IT IN NOW
     ,  OTHERWISE PRESS G": INPU
     T Y$
1007 IF Y$ <>"G" THEN DIM S$(208
     ): OPEN #4;"m";1;F$RND209:P
     OINT #4;VAL Y$: INPUT #4;S$
     : CLOSE #4: LET PTR=VAL Y$:
     CLS : GO TO 1021
1010 CLS : INPUT "TYPE IN THE NA
     ME OF ADDRESSEE:  "; LINE S
     $
1019 find S$,M(1),F$,PTR
1020 IF Y$="N" THEN PRINT AT 19,
     0;"NOT FOUND:PRESS ANY KEY
     FOR MENU": PAUSE 0: END PRO
     C
1021 DO
1023 RESTORE 5500: LET L=25
1024 FOR N=1 TO 8
1025 READ S,L$
1030 PRINT AT 20,0;L$;" ";S$(S T
     O S+L);"          ": PRINT AT
      9,0;"IF THIS IS NOT RIGHT
     PRESS 'N'": GET Y$
1040 IF Y$="N" THEN INPUT "NOW R
     ESUBMIT:
     "; LINE S$(S TO S+L)
1050 NEXT N
1051 CLS
1060 display# 2,S$,8
1100 PRINT AT 9,0;"IF THIS IS OK
      PRESS 'Y' TO FILE ": GET Y
     $: CLS
1110 LOOP WHILE Y$<>"Y"
1120 OPEN #4;"M";1;F$RND209
1130 POINT #4;PTR
1140 PRINT #4;S$
1150 CLOSE #4
1160 OPEN #4;"M";3;F$RND209
1170 POINT #4;PTR
1180 PRINT #4;S$
1190 CLOSE #4
1200 END PROC
2000 DEF PROC find REF S$, REF M
     (),F$, REF PTR
2005 LOCAL T$
2010 DIM T$(208)
2020 DO
2030 OPEN #4;"M";1;F$RND209: LET
      Y$="N"
2040 POINT #4;PTR: INPUT #4;T$
2044 CLOSE #4
2055 IF INSTRING(1,T$(1 TO 52),S
     $)<>0 THEN CLS :display# 2,
     T$,8: PRINT AT 9,0;"If this
      is the one you want     pr
     ess 'Y'": GET Y$: EXIT IF Y
     $="Y"
2085 LET PTR=PTR+1
2090 LOOP UNTIL PTR=M(1)+1
2095 IF Y$="Y" THEN LET S$=T$
2096 PRINT AT 9,0;"

     "
2100 END PROC
5000 DEF PROC display# K,S$,E
5010 RESTORE 5500
5020 FOR N=1 TO E
5030 READ S,H$: PRINT #K; S$(S T
     O S+25)
5040 NEXT N
5045 IF E=7 THEN PRINT #K;" ": P
     RINT #K;" "
5050 END PROC
5500 DATA 1,"N1",27,"N2",53,"A1"
     ,79,"A2",105,"A3",131,"A4",
     157,"PC",183,"TE"
6000 DEF PROC exit F$
6010 CLS : PRINT AT 9,9;"FAREWEL
     L"
6026 NEW
6030 END PROC
8000 DEF PROC 1st REF M(),F$
8005 LOCAL Y$,K,P,A$
8010 CLS
8015 DIM A$(52): DIM J$(M(1),211
     )
8020 PRINT AT 9,0;"PRESS P FOR P
     RINTER,S FOR SCREEN"
8025 DO
8030 GET Y$
8040 IF Y$="P" THEN LET K=3
8050 IF Y$="S" THEN LET K=2
8055 LOOP UNTIL Y$="P" OR Y$="S"
8058 CLS
8059 CLS
8060 INPUT "NUMBER TO START (OR
     'A' FOR ALL:"; LINE U$
8062 IF U$<>"A" THEN INPUT "NUMB
     ER TO END:"; LINE I$: LET P
     =VAL U$: LET I=VAL I$:
8064 IF U$="A" THEN GO TO 8132
8070 DO
8080 OPEN #4;"m";1;F$RND209
8090 POINT #4;P
8100 INPUT #4;A$
8110 CLOSE #4
8120 PRINT #K;P;" ";A$(1 TO 25),
     A$(26 TO 52)
8125 LET P=P+1
8130 LOOP UNTIL P>I
8131 GO TO 8140
8132 FOR N=1 TO M(1)
8133 OPEN #4;"M";1;F$RND209:POIN
     T #4;N: INPUT #4;J$(N): LET
      J$(N,209 TO 211)=STR$ N: C
     LOSE #4: NEXT N
8134 SORT J$()(27 TO 52): FOR N=
     1 TO M(1): PRINT #K; USING
     "###";VAL (J$(N,209 TO 211)
     );J$(N,1 TO 52);J$(N,183 TO
      203): NEXT N
8140 INPUT "PRESS ENTER FOR MENU
     ."; LINE Y$
8150 CLS
8160 END PROC
```



DISCIPLE DATA



John Wase with more advice for Disciple owners.

Having accepted the invitation to visit Rockfort products, one recent cold Monday found me in Hendon. There they were, fiddling with monitor and Spectrum, sat atop a printed circuit board. Yes, this was it; version 3 of the Disciple is now with us. Whilst I was able to see many of its features on the spot, this was clearly not a fully working system, and I was pleased to get an EPROM and revised software cassette through the post just in time to complete this article.

So, what's different? The only essential difference in the hardware are the snapshot and enable switches, on the left of the interface, which are now replaced with a new design, much stronger. Apart from this, there is the new EPROM and new cassette software.

The first and most obvious advantage is that the snapshot button now has three additional uses. Previously, on version 2, one could transfer 48K programs from tape or microdrive to disc, and dump a screen to the printer. With version 3 things get a bit complicated — you need to press a key at the same time as the snapshot button — but the rewards are there all right. You can save a full 128K program to disc — and reload it in a mere seven seconds. This disc interface is *very* fast. You can also save a single screen to disc. And, finally, you can send a double-size screen dump to the printer.

Apart from these games-oriented uses, there are further features for the more serious programmer. The most obvious are the direct access from BASIC to streams and channels in the same way as permitted by Interface 1 or Discovery. This, in turn, provides a variety of BASIC facilities; for instance, direct data filing, using sequential files, a considerable step forward. I know that comparable facilities were available in earlier versions, but only from machine code using Disciple's command codes (similar to Interface 1's hook codes which they emulate). EPROM version 3 also extends very considerably the number of command codes. Another new feature concerns an area of Disciple's RAM which can now be used by the serious programmer: a 512 byte machine code routine can be loaded directly into disc buffer and executed directly from there without using any of the Spectrum's memory. A particularly useful feature is the ability to "trap"Disciple's error messages. If, for instance, the disc filled up during an operation, one would normally get an error message indicating this and the program would stop. This new facility enables a jump back to BASIC, perhaps with an appropriate message. Code files, too, can be made autorunning by a third specifier after "CODE", giving the autorun address. There are new **POKE @** commands, an improved **COPY** command and lots of other programming features.

Network

In addition, the network facilities have now been much revised and extended. There is provision for "assistant" machines in addition to the "master", and these also can have their own drives. "Snapshotted" files can now be transferred across the network, and the master can send messages to slave machines whilst a program is being executed.

"Great," you might say. "But what about me; I've already got a Disciple." Well, for those not already in the know, there is a user club run by Bob Brenchley, 34, Bourton Road, Tuffley, Gloucester, GL4 OLE. This provides a helpline, programming tips and will also advise on upgrading to version 3: this can readily be achieved by changing the ROM and software.

Next, an apology. In my review of Disciple, I queried the amount of disc memory available, on the basis that there was no compacting facility. I was very rapidly told that this is unnecessary, since Disciple's directory is bit mapped, and therefore all free sectors can be used, irrespective of their physical location on the disc.

Now for some programming notes. Many microdrive users have asked about compatibility; what microdrive programs will run on Disciple!? The answer is "most". The reason is that the standard Interface 1 hook codes have been used in Disciple. Provided that the program uses these and these only, it will work. So the microdrive version of Devpac assemblers and monitors and of the Artist are fine. So are Masterfile, Omnicalc, Word manager, The Last Word and Icon Graphics 128. But Tasword 3 and Tasword 128 won't, although they will run with the programs on microdrives and the files on discs. The reason is that the program interacts too much with interface 1 ROM, evading the normal hook codes. For instance, the first thing it does is to check the version of interface 1 being used, so, naturally, it fails there.

SHORT CUTS

More prize winning programming from ZX readers.

Short Cuts exists as a showcase for the programming talent of ZX readers. Entries are welcome that show both the potential of the Spectrum and the expertise of the programmer.

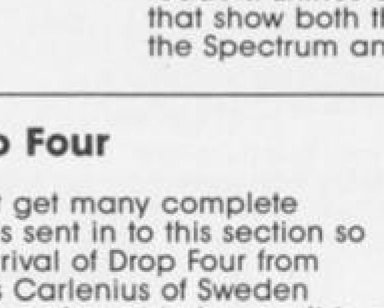
Submissions for Short Cuts must be your own original programming and not have been published previously.

Drop Four

I don't get many complete games sent in to this section so the arrival of Drop Four from Niclas Carlenius of Sweden made a pleasant change. It is a two player game based on a well known counter frame game where the idea is to get four counters of your colour/symbol in a straight line, left/right, up/down or diagonally.

It is neat and compact and has nice screen presentation. The check for win logic is kept simple and for ardent programmers there is the challenge of adding routines to make the computer play against you intelligently.

It is not a novel program, but we liked it and make it this month's Star Cut.




```
   1 REM Drop four
 100  LET x=1: LET i$="XXX": INK
 0: PAPER 7: CLS : DIM a(12,13)
 110 LET s$=CHR$ 144+CHR$ 145+CH
R$ 146
 120 LET t$=CHR$ 147+CHR$ 95+CHR
$ 148
 130 LET u$="                  "
 140 DATA 0,127,127,127,127,127,
127,127
 150 DATA 0,255,255,255,255,255,
255,255
 160 DATA 0,254,254,254,254,254,
254,254
 170 DATA 128,128,128,128,128,12
8,128,255
 180 DATA 1,1,1,1,1,1,1,255
 190 FOR j=144 TO 148: FOR k=0 T
O 7: READ a: POKE USR CHR$ j+k,a
: NEXT k: NEXT j
 220 FOR j=2 TO 17 STEP 3: FOR k
=2 TO 26 STEP 4
 240 PRINT  INK 6; PAPER 0;AT j,
k;s$
 250 PRINT  PAPER 6;AT j+1,k;t$:
 NEXT k: NEXT j
 270 PRINT  INK 3;AT 0,3;"1   2
  3   4   5   6   7"
 280 PRINT  INK 7; PAPER x;AT 20
,11;"PLAYER ";x
 281 LET d=0: FOR b=1 TO 7
 283 IF ATTR (2,(b-1)*4+2)<>6 TH
EN  LET d=d+1
 284 NEXT b
 285 IF d=7 THEN  CLS : DIM a(12
,13): GO TO 220
 290 INPUT "Which column? (1 to
7) ";a
 300 IF a<>INT a OR a>7 OR a<=0
THEN  GO TO 290
 310 LET c=(a-1)*4+2
 320 IF ATTR (2,c)<>6 THEN  PRIN
T  FLASH 1;AT 21,7;"Column ";a;"
 is full": GO TO 290
 330 PRINT AT 21,7;u$
 340 FOR j=0 TO 18: LET z=ATTR (
j,c): PRINT AT j,c; PAPER x; INK
 7;i$
 370 IF j=2 THEN  PRINT  INK 3;A
T j-2,c;" ";a;" "
 380 IF j>2 AND z=6 THEN  PRINT
 PAPER 6;AT j-2,c;t$
 390 IF INT (j/3)=j/3 THEN  PRIN
T AT j-2,c;"   "
 400 IF j>2 AND z=56 THEN  PRINT
  INK 6; PAPER 0;AT j-2,c;s$
 410 IF j=18 OR ATTR (j+2,c)=23
OR ATTR (j+2,c)=15 THEN  BEEP .1
,-20: GO TO 440
 420 IF z=48 THEN  FOR p=1 TO 4:
 BEEP .05,(20-2*j)+p: NEXT p
 430 NEXT j
 440 LET l=j/3+3: LET c=a+3: LET
 a(l,c)=x
 460 FOR j=-3 TO 0
 470 IF a(l+j,c)=x AND a(l+j+1,c
)=x AND a(l+j+2,c)=x AND a(l+j+3
,c)=x THEN  GO TO 550
 480 IF a(l,c+j)=x AND a(l,c+j+1
)=x AND a(l,c+j+2)=x AND a(l,c+j
+3)=x THEN  GO TO 550
 490 IF a(l+j,c+j)=x AND a(l+j+1
,c+j+1)=x AND a(l+j+2,c+j+2)=x A
ND a(l+j+3,c+j+3)=x THEN  GO TO
550
 500 IF a(l-j,c+j)=x AND a((l-j)
-1,c+j+1)=x AND a((l-j)-2,c+j+2)
=x AND a((l-j)-3,c+j+3)=x THEN
GO TO 550
 510 NEXT j
 520 LET i$="OOO": IF x=2 THEN
LET i$="XXX"
 530 LET x=3-x: GO TO 280
 550 PRINT  OVER 1; FLASH 1;AT 2
0,11;u$( TO 8)
 560 PRINT AT 20,9; FLASH 1; INK
 x;"Player ";x;" Wins"
 570 INPUT "Another game ? ";a$:
 IF a$(1)<>"y" THEN  STOP
 580 CLS : DIM a(12,13): GO TO 2
20
```

Vector Table

Vector Table is one of those seemingly pointless routines which many of us find so fascinating. Once entered and run a machine code routine is set up at 64250+

To start it type RANDOMIZE USR 64319 and to cancel it type RANDOMIZE USR 64337, make sure that no code is placed above 62000 or else the machine will crash.

So what does it do?

Well after starting it you will find that pressing keys 0 and 9 at the same time will clear the screen and leave it blank, to bring back the display press key q.

Perhaps useful for those who suffer from obsessive secrecy and who do not want others to see what they are doing on the computer? Useful for BBC programme producers?

Our thanks to Hugh Shaw of Ayreshire for this one.

```
9000 REM vector table version
9025 CLEAR 62000
9030 FOR j=64250 TO 64343
9035 READ a : POKE j,a
9040 NEXT j
9050 DATA 197,229,213,245,221,22
9,62,239
9060 DATA 219,254,246,224,254,25
2,32,44
9070 DATA 1,0,3,33,0,88,17,248
9080 DATA 247,229,197,237,176,19
3,225,84
9090 DATA 30,1,62,63,119,237,176
,62
9100 DATA 251,219,254,246,224,25
4,254,32
9110 DATA 246,1,0,3,33,248,247,1
7
9120 DATA 0,88,237,176,221,225,2
41,209
9130 DATA 225,193,195,56,0,33,0,
254
9140 DATA 1,250,0,113,35,16,252,
113
9150 DATA 62,254,237,71,237,94,2
01,62
9160 DATA 62,237,86,237,71,201
```

Printer Chars

ZX Printer Chars is a program which uses a ZX, Alphacom or similar printer to print in various sizes and was submitted by Tom Vandepoel from Belgium.

The routine asks for the number of chars high you want and then for the number of chars wide. This latter number cannot be greater than 4.

This could be a starting point for several print routines including centering etc.

```
  1 REM ZX Printer chars.
 10 INPUT "Text to be printed "
;a$: INPUT "Height/point ";h: IN
PUT "Width/point ";w
 15 FOR n=1 TO LEN a$: LET h$="
": LET s$="": PRINT AT 21,0;a$(n
)
 17 FOR e=1 TO h: LET h$=h$+a$(
n): LET s$=s$+" ": NEXT e
 20 FOR x=0 TO 7: LET p$="": FO
R y=0 TO 7
 30 LET p$=p$+(h$ AND POINT (x,
y)=1)+(s$ AND POINT (x,y)=0)
 40 NEXT y: FOR e=1 TO w: LPRIN
T p$: NEXT e: NEXT x: NEXT n
```

Highscore

David Knight is a reader from Didcot who has featured often in these pages and we welcome his routine for inclusion in your own game programs.

Highscore is a routine for those of you who cannot work out (or can't be bothered to work out) a means of adding a chart of high scores. This simple routine works well but needs the addition of four variables in a program, at the initialisation of variables stage add LET no=the number of names to be held DIM h$(no,10): DIM h(no) to your lines.

In the game itself the score should be kept in variable 'sc' and at the end of the game a line should include GO SUB 9000 — Happy zapping...

```
8999 STOP
9000 REM  highscore
9010 REM no=number of names
         sc=player's score
  h$(no,10)=table of names
      h(no)=table of scores.
9020 LET a=1
9030 IF h(a)<sc THEN  GO TO 9060
9040 LET a=a+1
9050 IF a<=no THEN  GO TO 9030
9060 IF a>no THEN  PRINT "You sc
ored: ";sc;"."'"Not a high score
 !": PAUSE 100: GO TO 9090
9070 FOR b=no-1 TO a STEP -1: LE
T h$(b+1)=h$(b): LET h(b+1)=h(b)
: NEXT b
9080 INPUT "Well done, a high sc
ore !"'"Your name ? ";h$(a): LET
 h(a)=sc
9090 CLS : PRINT AT 0,0;"High Sc
ores"
9100 PRINT
9110 FOR a=1 TO no: PRINT h$(a),
h(a): NEXT a
9120 PAUSE 0: CLS : RETURN
```

Telephone Book

A simple little database program for the specific purpose of keeping a list of names and phone numbers. Perhaps its a little pointless on a tape based system — it would be quicker to look them up in a traditional book — but for disc or microdrive owners it could be of real use.

An interesting system of adding another number is employed, most of us would set up the file with a predetermined maximum number of entries, Lawrence Miller who wrote it, transfers data to a temporary array, re-DIMs the permanent store one element greater, and then refills it from the temporary store.

Perhaps with a larger file system the waiting time would be unacceptably long, but for the small number of entries this is designed to hold it works a treat. This also means that saving and loading time is kept to a minimum.

I would suggest a modification to the saving method so that only the required data was saved ie. SAVE name$ DATA a$() then a further option to LOAD data (by LOAD name$ DATA a$() would allow even faster and more economical storage and also the ability to store and retrieve more than one set of numbers.

I leave you to experiment...

```
  5 LET z=1: DIM a$(z,32): LET
c$=" ": DIM e$(1,15): LET f$="
                              "
 10 POKE 23658,8: CLS : PRINT "
TELEPHONE DIRECTORY"''"1> ENTER
NEW FILE"'"2> VIEW FILES"'"3> DE
LETE/EDIT FILE"'"4> SAVE/VERIFY"
 20 PAUSE 0: LET x=CODE INKEY$:
 IF x<49 OR x>52 THEN  GO TO 20
 30 CLS : GO SUB 100*(x-48)
 40 GO TO 10
100 PRINT AT 10,10; FLASH 1;"PL
EASE WAIT": FOR n=1 TO z: IF a$(
n,1)=" " THEN  LET q=n: GO TO 14
0
110 NEXT n
120 DIM z$(z,32): FOR n=1 TO z:
 LET z$(n, TO 32)=a$(n, TO 32):
NEXT n
130 LET z=z+1: LET q=z: DIM a$(
z,32): FOR n=1 TO z-1: LET a$(n,
 TO 32)=z$(n, TO 32): NEXT n
140 CLS : INPUT "ENTER NAME (MA
X. 15 Chrs.)"'"OR '0' FOR MENU"'
a$(q, TO 15): IF a$(q,1)="0" THE
N  LET a$(q,1)=" ": RETURN
150 INPUT "NATIONAL CODE"'a$(q,
17 TO ): INPUT "EXCHANGE CODE"'a
$(q,22 TO ): INPUT "REMAINING NU
MBERS"'a$(q,26 TO )
160 GO SUB 240: PRINT a$(q): IN
PUT "IS THE FILE CORRECT Y/N"'b$
170 IF b$="N" THEN  GO TO 140
180 IF b$="Y" THEN  RETURN
190 GO TO 160
200 INPUT "ENTER FIRST LETTER O
F FILE (A-Z)OR '0' FOR MENU"'c$(
 TO 1): IF c$="0" THEN  RETURN
210 LET c=0: GO SUB 240: FOR n=
1 TO z: IF a$(n,1)=c$ THEN  PRIN
T 'a$(n): LET c=c+1
220 IF c=9 THEN  LET c=0: GO SU
B 390: GO SUB 240
230 NEXT n: GO TO 200
240 CLS : PRINT  INVERSE 1;"FIL
E NAME","PHONE NUMBER",': RETURN

300 INPUT "ENTER FILE NAME FOR
DELETION OR EDITING OR ('0' FOR
MENU)"'e$(1): IF e$(1,1)="0" OR
e$(1,1)=" " THEN  RETURN
310 FOR n=1 TO z: IF a$(n, TO 1
5)=e$(1) THEN  GO SUB 240: PRINT
 a$(n): GO TO 340
320 NEXT n
330 GO SUB 380: CLS : GO TO 300
340 INPUT "ENTER ('D' DELETE)('
E' EDIT)"'"OR '0' FOR MENU"'c$:
IF c$="0" THEN  RETURN
350 IF c$="E" THEN  LET q=n: GO
 TO 140
360 IF c$="D" THEN  LET a$(n)=f
$: RETURN
370 GO TO 340
380 PRINT AT 10,10; FLASH 1;"NO
T IN FILE"
390 PRINT 'TAB 10; FLASH 1;"PRE
SS A KEY": PAUSE 0: RETURN
400 DIM z$(1,1): BEEP .2,20: SA
VE "phonenum." LINE 10: PRINT "V
ERIFY PROG.": VERIFY "": RETURN
```

Memory Usage

Probably the shortest ever shortcut, this one line program is useful for adding to your own programs under development. By entering GO TO 9999 every so often you will get a constant up date of the free memory, length of your program and the space the variables take up. For the last feature to operate your program needs to have been RUN first.

Mr. D.Mcmillan of Glasgow devised this routine.

```
9999 DEF FN a(a$)=VAL ("PEEK "+a
$+"+256*PEEK ("+a$+"+1)"): PRINT
 "F=";FN a("23730")-FN a("23641"
)'"P=";FN a("23627")-FN a("23635
")'"V=";FN a("23641")-FN a("2362
7")-SGN PI
```

PRINTER TOOLKIT

The Spectrum 128 RS232 printer port is mastered by S. J. Nutting.

There are three routines in this toolkit, all of which are designed to make the operation of a printer via the 128's RS232 port much easier than it currently is. The printer used was the Brother M1009 which is fairly Epsom standard and, where necessary, the ability to modify the program for other printer codes is included.

Screen Copier

Type in the program and run it. Answer the prompts, either by pressing ENTER to accept the set

Screen Copier

```
 10 CLEAR 65099: LET c=0:  FOR
a=65100 TO 65332: READ n: POKE a
,n: LET c=c+n: NEXT a
 20 IF c<>27505 THEN  PRINT "DA
TA ERROR": STOP
 30 DATA 205,207,254,33,53,255,
205,18,255,33,56,255,205,18,255,
58,39,255,111,38,0,41,41,41,41,3
4,41,255,33,37,255,229,205,222,2
54,225,205,243,254
 40 DATA 33,62,255,205,18,255,3
3,41,255,126,205,27,255,35,126,2
05,27,255,58,39,255,245,62,134,2
22,8,50,160,254,254,62,40,32,245
,175,42,43,255
 50 DATA 221,33,45,255,6,8,203,
126,40,3,221,134,0,221,35,36,16,
244,205,27,255,205,27,255,241,24
,215,42,43,255,35,34,43,255,241,
61,32,201,33,59,255
 60 DATA 205,18,255,33,37,255,5
2,33,40,255,53,32,153,33,0,88,1,
0,3,203,190,35,11,120,177,200,24
,247,70,35,78,120,230,248,198,64
,103,120,230,7,15,15,15
 70 DATA 129,111,34,43,255,201,
70,35,78,120,203,47,203,47,203,4
7,198,88,103,120,230,7,15,15,15,
129,111,58,39,255,203,254,35,61,
32,250,201,70,35,126,205,27,255,
16,249,201,229,197,245,205,66,25
5,241,193,225,201,0,0,32,24,0,0,
0,0,128,64,32,16,8,4,2,1
 80 RESTORE 130: LET add=65333:
 LET e$="Press ENTER after each
byte has been inputed pressing j
ust ENTERwill go on to the next
set of   inputs"
 90 CLS : READ memory: READ t$:
 IF memory=0 THEN  POKE add-a,0:
 PRINT "SAVE MACHINE CODE with"'
"SAVE""mc"" CODE 65100,";add-651
00: STOP
 100 PRINT AT 21,0;: RANDOMIZE a
dd: POKE  memory,PEEK 23670: POK
E memory+1,PEEK 23671
 110 PRINT AT 16,0;e$'+t$: FOR a
=1 TO 50: INPUT b$: IF b$<>"" TH
EN  POKE add+a,VAL b$: PRINT b$;
" ";: NEXT a
 120 POKE add,a-1: LET add=add+a
: GO TO 90
 130 DATA 65104
 140 DATA "Initalise printer cod
es 27 64"
 150 DATA 65110
 160 DATA "Line feeds codes 27 4
9"
 170 DATA 65216
 180 DATA "Carriage return codes
 10 13"
 190 DATA 65140
 200 DATA "Bit Image codes 27 42
 1"
 210 DATA 65311
 220 DATA "Output A reg routine"
+CHR$ 13+"end with 201 (RET)"
 230 DATA 0," "
```

codes or type in your own (one at a time) and the program will set up the code at address 65100. You will need to reload it before you can make screen copies.

Not only can you copy the whole screen but also sections of your choice. Before using the routine you will have to set up the screen area required by POKEing the following addresses.

POKE 65317,y
POKE 65318,x
POKE 65319,chars_across
POKE 65320,chars_down

The x,y,chars_across and chars_down variables need to be replaced by numbers representing:

y — the starting position *down* the screen
x — the starting position *across* the screen
chars_across — the number of characters across the screen that you want printed
chars_down — the number of lines down the screen from the start position that you wish to print.
EG. To set up a whole screen: x=0:y=0:chars_across=32:Chars_down=24 (see Fig. 1).

Graphitext

This routine provides a means of producing graphics, UDGs and user defined character sets. These can be used either from a BASIC string variable or from within a word processor.

You will have to create whatever UDGs you wish to use and store them from 57320 in blocks of 768 bytes (you have room for ten sets).

Type in the program and make any alternative entries that you require to suit your printer and SAVE the code as instructed. Make sure it has been loaded back before using it!

From BASIC, set up the variable a$ with the text, graphics etc. that you wish to print and call the routine with **RANDOMIZE USR 65005.**

From a word processor, SAVE the text, exit the program, load this program, reload the text and **RANDOMIZE USR 65000.** Text is set up for the LAST WORD text address and if you are using that program then there is no need to make any adjustments, however the bytes **65001** and **65002** hold the address of the start of the text and if you are using TASWORD 2 or another word-pro then you will need to POKE the start address to them, i.e. for TASWORD 2 (text file stored at 32000)

POKE 65001,INT(32000/256)
POKE 65002,32000-256*INT(32000/256)

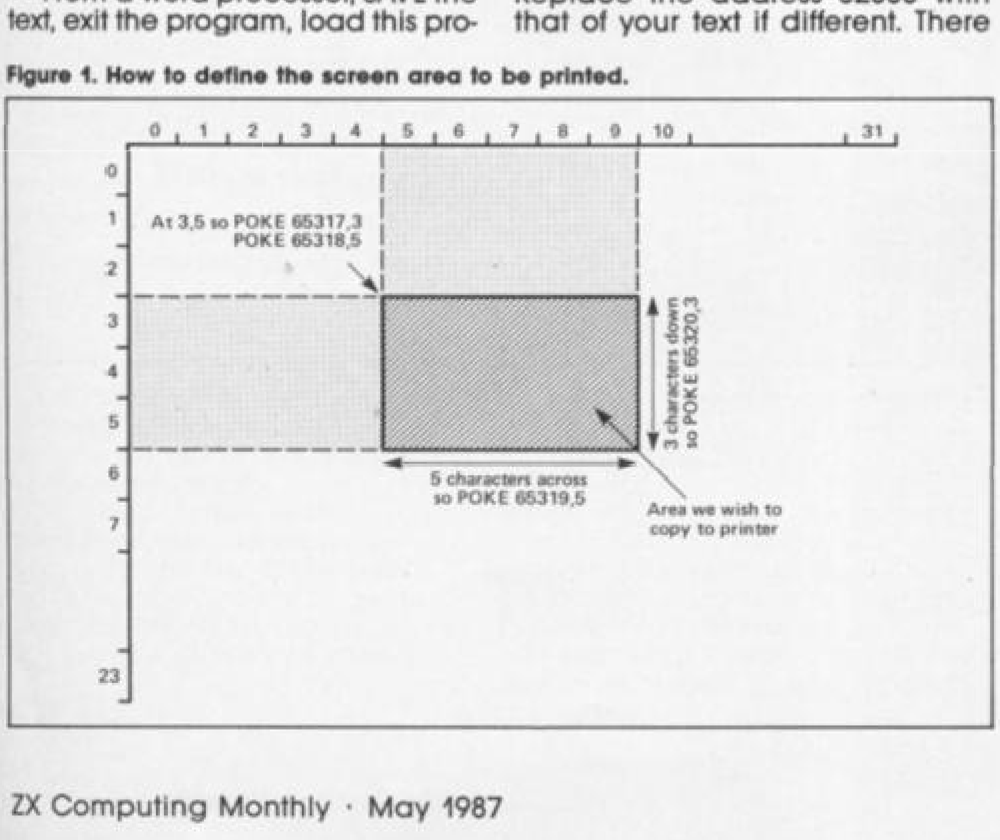
Replace the address 32000 with that of your text if different. There are a couple of rules which you MUST follow. To change between character sets you need to set up some of the control code characters. The routine defaults to set 1 at 57320, but to change it the codes from 128 to 137 should be sent embedded in the text.

Finally, a control code should be defined as 255 and placed at the end of the text. This is the routine's instruction to cease printing (end of text marker).

Graphitext program

```
  10 CLEAR 64999: PRINT "STOP TA
PE"'"PLEASE WAIT": LET c=0: FOR
a=65000 TO 65305: READ n: LET c=
c+n: POKE a,n: NEXT a
  20 IF c<>37507 THEN  PRINT "DA
TA ERROR": STOP
  30 DATA 33,50,117,24,26,42,75,
92,126,254,128,32,5,205,243,254,
207,1,254,65,40,6,205,184,25,235
,24,236,35,35,35,229,33,232,222,
34,54,92,175
  40 DATA 50,25,255,33,26,255,20
5,251,254,225,229,175,50,24,255,
126,254,13,40,32,254,14,40,28,25
4,255,202,243,254,254,128,48,11,
58,24,255
  50 DATA 254,120,40,7,60,50,24,
255,35,24,224,205,243,254,207,15
,225,58,24,255,254,0,40,83,229,1
11,38,0,41,41,41,229,33,39,255,2
05,251,254
  60 DATA 225,125,205,4,255,124,
205,4,255,225,126,254,13,40,54,2
54,14,40,50,254,128,40,79,254,12
9,48,24,229,78,6,0,42,54,92,62,8
,9,61,32,252
  70 DATA 205,201,254,33,24,255,
53,225,35,24,215,229,203,191,71,
33,232,219,17,0,3,25,16,253,34,5
4,92,24,234,229,58,25,255,254,0,
32,17,33,29,255
  80 DATA 205,251,254,33,36,255,
205,251,254,225,35,195,25,254,33
,33,255,205,251,254,24,237,58,25
,255,238,1,50,25,255,24,192,34,2
2,255,62,134
  90 DATA 222,8,50,226,254,254,6
2,200,245,175,42,22,255,221,33,1
4,255,6,8,203,126,40,3,221,134,0
,221,35,35,16,244,205,4,255,241,
24,219,225
 100 DATA 33,0,60,34,54,92,201,7
0,35,126,205,4,255,16,249,201,22
9,197,245,205,43,255,241,193,225
,201,128,64,32,16,8,4,2,1,0,0,0,
0
 110 RESTORE 160: LET add=65306:
 LET e$="Press ENTER after each
byte has been inputed pressing j
ust ENTERwill go on to the next
set of   inputs"
 120 CLS : READ memory: READ t$:
 IF memory=0 THEN  POKE add-a,0:
 PRINT "SAVE MACHINE CODE with"'
"SAVE""mc"" CODE 65000,";add-650
00: STOP
 130 PRINT AT 21,0;: RANDOMIZE a
dd: POKE  memory,PEEK 23670: POK
E memory+1,PEEK 23671
 140 PRINT AT 16,0;e$'+t$: FOR a
=1 TO 50: INPUT b$: IF b$<>"" TH
EN  POKE add+a,VAL b$: PRINT b$;
" ";: NEXT a
 150 POKE add,a-1: LET add=add+a
: GO TO 120
 160 DATA 65043
 170 DATA "Initalise printer cod
es 27 64"
 180 DATA 65191
 190 DATA "Feed pitch code for s
pacing between lines 27 51 38"
 200 DATA 65208
 210 DATA "Close line feed 27 49
"
 220 DATA 65197
 230 DATA "Carriage return codes
 10 13"
 240 DATA 65108
 250 DATA "Bit Image codes 27 42
 1"
 260 DATA 65288
 270 DATA "Output A reg routine"
+CHR$ 13+"end with 201 (RET)"
 280 DATA 0," "
9999 CLEAR : SAVE "textprint" LI
NE 40
```

Figure 1. How to define the screen area to be printed.



Printcopy

This is a short, relocatable routine to print non-graphic characters, thus cutting out the need to spend a lot of time doing a bit-image screen copy.

The kind of application may be to print PRESTEL screens when the graphics are not required but the text is.

Enter the program and run it. Enter the address you wish to locate the code at and then save it with SAVE "code" address,37. Simply call the address to use it.

```
  10 INPUT "Start Address:";S: L
ET c=0: FOR s=s TO s+36: READ n:
 POKE s,n: LET c=c+n: NEXT s
  20 IF c<>3504 THEN  PRINT "DAT
A ERROR": STOP
  30 DATA 62,3,205,1,22,1,0,0,12
0,254,32,40,12,197,205,56,37,215
,205,197,22,193,4,24,239,121,254
,24,200,12,6,0,62,13,215,24,227
```

SPECTRUM PROGRAMMING

EXTENDED BASIC

Stuart Nicholls demonstrates how to add some new commands to Spectrum Basic.

I suppose that most of you have at some time or another wished that there were more commands available in ZX BASIC. Commands such as BOX or FILL maybe. Well, with this article I hope to show the competent machine code programmer how to achieve this.

Yes you've guessed it, another BASIC extension article. But before you stop reading I believe that this one is different from any other you may have read in that it does not use PRINT or LPRINT to access the new commands, and more importantly, works without Interface I.

I will be explaining how you can add 49 new commands that will be checked for syntax before entry into the listings and checked for parameter errors in run time in *exactly* the same way as normal ZX Spectrum commands.

In order to understand how new commands can be added to ZX BASIC we must first examine how the Spectrum handles the existing 50 commands, that is commands DEF FN (code CEh) to COPY (cod FFh).

Commands

When the Spectrum is first switched on it goes through an initial setting up procedure to check the RAM available and give initial values to the system variables.

Once this has been carried out we enter what is known as the 'main execution loop' which allows commands to be entered either as a program listing or as direct commands to be executed. But before a line can be accepted a check is made to see if the commands within the line conform to the correct syntax. If there is any error in the syntax then this is highlighted with a question mark at the position of the error so that it can be corrected. Only when the syntax is correct will the line be either added to the listing or RUN. There may of course still be parameter errors in the line which will cause run-time errors stopping the program with an error report.

Listing 1

```
** pass          1
** errors        0
** warnings      0
                 249 *LLIST    ON
F1EB             250 line.scan:
                     ;             CHECK SYNTAX OF LINE.
                     ;             CHECK SYNTAX OF LINE.
F1EB FDCB01BE            RES  7,(IY+1)
F1EF CDFB19              CALL #19FB
F1F2 AF                  XOR  A
F1F3 32475C              LD   (23623),A
F1F6 3D                  DEC  A
F1F7 323A5C              LD   (23610),A
F1FA 1801                JR   loop.1
F1FC E7          260 statement.loop:
                         RST  #20 ;    ADVANCE CH.ADD.
F1FD CDBF16      270 loop.1:
                         CALL #16BF ;CHECK EACH STATEMENT TO END OF LINE.
F200 FD340D              INC  (IY+13)
F203 FA8A1C              JP   M,#1C8A
F206 DF                  RST  #18
F207 0600                LD   B,0
F209 FE0D                CP   #D
F20B CAC2F2              JP   Z,line.end
F20E FE3A                CP   #3A
F210 28EA                JR   Z,statement.loop
F212 2185F2              LD   HL,statement.ret
F215 E5                  PUSH HL
F216 4F                  LD   C,A ;CHECK COMMAND AT BEGINING OF STATEMENT.
F217 E7                  RST  #20
F218 219EF3              LD   HL,.A ;SET HL TO START OF NEW OFFSET TABLE.
F21B 79                  LD   A,C
F21C FE2E                CP   46 ;    IS IT A DOT COMMAND?
F21E 281A                JR   Z,loop.4
F220 FEF4                CP   244 ;   DIVERT REM/POKE/IF COMMANDS.
F222 2808                JR   Z,loop.2
F224 FEFA                CP   250
F226 2807                JR   Z,loop.2
F228 FEEA                CP   234
F22A 2803                JR   Z,loop.2
F22C 21481A              LD   HL,#1A48 ;SET HL TO START OF ROM OFFSET TABLE.
F22F D6CE        280 loop.2:
                         SUB  #CE ;    FOR STANDARD SPECTRUM COMMANDS.
F231 DA8A1C              JP   C,#1C8A
F234 4F          290 loop.3:
                         LD   C,A ;   GET COMMAND OFFSET.
F235 09                  ADD  HL,BC
F236 4E                  LD   C,(HL)
F237 09                  ADD  HL,BC ;USE OFFSET TO POINT TO PARAMETER TABLE.
F238 1823                JR   get.parameter ;EITHER NEW OR NORMAL.
F23A DF          300 loop.4:
                         RST  #18 ; ACCEPT EITHER UPPER OR LOWER CASE
F23B 4F                  LD   C,A ; LETTER DOT COMMANDS.
F23C E7                  RST  #20
F23D 79                  LD   A,C
F23E CBAF                RES  5,A
F240 FE5B                CP   91
F242 D28A1C              JP   NC,#1C8A
F245 D641                SUB  65
F247 DA8A1C              JP   C,#1C8A
F24A 79                  LD   A,C
F24B FE61                CP   97
F24D 3802                JR   C,loop.5
F24F D606                SUB  6
F251 D641            loop.5:
                         SUB  65
F253 219EF3              LD   HL,.A ; USE NEW OFFSET TABLE.
F256 0600                LD   B,0
F258 18DA                JR   loop.3
F25A 2A745C      310 scan.loop:
                         LD   HL,(23668)
F25D 7E          320 get.parameter:
                         LD   A,(HL) ;EXECUTE COMMANDS AND ALLOW FOR
F25E 23                  INC  HL ;      SEPARATORS.
F25F 22745C              LD   (23668),HL
F262 015AF2              LD   BC,scan.loop
F265 C5                  PUSH BC
F266 21011C              LD   HL,#1C01 ;USE ROM CLASS OFFSET TABLE FOR
F269 4F                  LD   C,A ;        CLASS 06 TO 0B
F26A FE20                CP   #20
F26C 3010                JR   NC,separator
F26E FE06                CP   6
F270 3003                JR   NC,get.2
F272 2110F3              LD   HL,class ;USE  NEW CLASS OFFSET FOR
                 325 ;                         CLASS 00 TO 05
F275 0600        330 get.2:LD   B,0 ;   COMPUTE START ADDRESS OF COMMAND
F277 09                  ADD  HL,BC ; CLASS ROUTINE.
F278 4E                  LD   C,(HL)
F279 09                  ADD  HL,BC
F27A E5          340 get.1:PUSH HL
F27B DF                  RST  #18
F27C 05                  DEC  B
F27D C9                  RET    ;AND JUMP TO IT.
F27E DF          350 separator:
                         RST  #18 ;IF SEPARATOR NOT PRESENT THEN GIVE
F27F B9                  CP   C ;  ERROR REPORT.
F280 C28A1C              JP   NZ,#1C8A
F283 E7                  RST  #20
F284 C9                  RET
                 355 *LLIST    OFF

** pass          2
** errors        0
** warnings      0
```

SPECTRUM (48K/+)

Listing 2

```
** pass         1
** errors       0
** warnings     0
                605 #LLIST    ON
                610 ;***** SYNTAX OFFSET TABLE *****
F39E 00         620 .A   :DB   0
F39F 00             .B   :DB   0
F3A0 00             .C   :DB   0
F3A1 00             .D   :DB   0
F3A2 00             .E   :DB   0
F3A3 00             .F   :DB   0
F3A4 00             .G   :DB   0
F3A5 00             .H   :DB   0
F3A6 00             .I   :DB   0
F3A7 00             .J   :DB   0
F3A8 00             .K   :DB   0
F3A9 00             .L   :DB   0
F3AA 00             .M   :DB   0
F3AB 00             .N   :DB   0
F3AC 00             .O   :DB   0
F3AD 00             .P   :DB   0
F3AE 00             .Q   :DB   0
F3AF 00             .R   :DB   0
F3B0 00             .S   :DB   0
F3B1 00             .T   :DB   0
F3B2 00             .U   :DB   0
F3B3 00             .V   :DB   0
F3B4 00             .W   :DB   0
F3B5 00             .X   :DB   0
F3B6 00             .Y   :DB   0
F3B7 00             .Z   :DB   0
F3B8 00         630 .a   :DB   0
F3B9 00             .b   :DB   0
F3BA 00             REM  :DB   0 ;or .c
F3BB 00             .d   :DB   0
F3BC 00             .e   :DB   0
F3BD 00             .f   :DB   0
F3BE 00             .g   :DB   0
F3BF 00             .h   :DB   0
F3C0 00             .i   :DB   0
F3C1 00             .j   :DB   0
F3C2 00             .k   :DB   0
F3C3 00             .l   :DB   0
F3C4 00             POKE :DB   0 ;or .m
F3C5 00             .n   :DB   0
F3C6 00             .o   :DB   0
F3C7 00             .p   :DB   0
F3C8 00             .q   :DB   0
F3C9 00             .r   :DB   0
F3CA 00             IF   :DB   0 ;or .s
F3CB 00             .t   :DB   0
F3CC 00             .u   :DB   0
F3CD 00             .v   :DB   0
F3CE 00             .w   :DB   0
F3CF 00             .x   :DB   0
F3D0 00             .y   :DB   0
F3D1 00             .z   :DB   0
                635 ;** DOT COMMAND PARAMETER TABLE **
F3D2 00         640      DB    0
F3D3 0000       650 pno.routine:
                         DW    0 ;ALLOW ALL DOT COMMANDS TO PASS SYNTAX CHECK.
F3D5 05         660      DB    5
F3D6 0000       670 prem :DW   0 ;REM COMMAND CLASS 05
F3D8 0000       680      DB    0,0
F3DA 0000       690 ppoke:DW   0 ;POKE COMMAND CLASS 08 00
F3DC 06CB05     700      DB    6,#CB,5
F3DF 0000       710 pif  :DW   0 ;IF COMMAND CLASS 06 THEN 05
F3E1 093B0A3B
     0800       720      DB    #9,#3B,#A,#3B,#8,#0
F3E7 0000       730 ptext:DW   0 ;NEW DOT A COMMAND CLASS 09;0A;08 00
F3E9 20062905   740      DB    #20,#6,#29,#5
F3ED 0000       750 pinput:
                         DW    0 ;NEW INPUT COMMAND CLASS (06) 05
F3EF CF         760 no.routine:
                         RST   #8 ;ERROR REPORT FOR ALL REMAINING DOT COMMANDS
F3F0 0B                  DB    #B
                762 #LLIST    OFF

** pass         2
** errors       0
** warnings     0
```

With this generalised view of the Basic Interpreter in mind we can proceed to analyse each step in more detail.

It is easier to follow the workings of the interpreter by using a simple BASIC program and follow it stage by stage as it is interpreted, so for our example I will use the direct command:

POKE 65535, 8

As outlined above, the first step is to get the line into the Spectrum.

This is done by means of the editor which is called after the initialisation routine. The function of the editor is to 'read' key presses and build up a line in the editing area and also echo the line to the screen so that we can see what is being typed. The lower screen is used for this purpose.

So, if we type **POKE** (keyword key 0) **65535,8** this will appear on the screen and in the editing area. We can continue to type as much as we like, although the Spectrum will complain if the line is too long, and the editor will continue to accept key presses until ENTER is pressed.

As soon as ENTER is pressed a return from the editor is made, the basic interpreter is called and the line is 'run'. This is, however, a dummy run to check the syntax of the line in the editing area, only when the line passes this check is it run again to perform the command (assuming a direct command).

There are nine main steps taken during syntax checking:

1. Reset bit 7, flags to signal syntax checking.
2. Check the first byte of the statement for one of the command codes, CEh to FFh, and if OK move to step 3, otherwise signal an editor error.
3. Subtract CEh from code.
4. Add the result of '3' to the base address of an offset table.
5. Use the offset value found at this computed address and add this value to the computed address.
6. The result of '5' produces an address in the command class parameters table.
7. Use the address produced by '6' as the start to a series of class codes.
8. Work through each class code in turn using a class code offset table to call the class code routines.
9. If each class code routine is passed with no errors then move on the to the next statement in the editing area.

From this more detailed breakdown above you will realise that the syntax of each command depends on its class codes, and hence the class code routines. **Table 2** gives the method of calculating the class routine from a class code and **table 3** gives a brief description of the function of the class routines.

For our purposes it is not necessary to go into how the class routines work but it is essential to understand the result of each class routine. We can forget Class 01, 02, 07 and 0B routines as these are specific to certain commands and are of little use to us. The remaining class routines, except one (05), are concerned with evaluation of the expression following the command code, be it a numeric or string expression (for example a$, "help", a, 900, (a=b), NOT c, SIN x). Providing the expression

conforms to the requirements given in table 3 the syntax will be correct.

Coming back to the example **POKE 65535,8** the syntax checking will arrive at class code 08 and 00 (followed by a command routine address 1E80h) the route followed is given in diagram 1. Looking at **table 3** shows that Class 08 will, during syntax, check for two numeric expressions seperated by a comma. The syntax check will then move on to the class 00 routine which simply means that there must be no more expressions before the end of statement/line marker, and a return is made in this case to the main execution loop.

If, however, we had typed in **POKE 65535;8** the class 08 routine would have spotted the syntax error of the semicolon in place of the comma and returned to the editor with a '?' after the semicolon. Similarly, **POKE "help",8** would give a class 08 error as the first expression is a string, but **POKE a,b** would be accepted since a and b are numeric variables and in syntax checking the actual values are not calculated.

Our command has now passed its syntax check and a return made to the main execution loop. The next step is to RUN the command. This involves the same process as for syntax checking except that BIT 7, FLAGS is SET to indicate 'RUN TIME', and we enter the interpreter at a slightly different point. However this time after performing class routines 08 and 00 the actual values of the two numeric expressions will be placed on the calculator stack. We then go on to call the command routine given by the two byte number following the class codes in the parameter table, ie. address 1E80h.

Listing 3

```
** pass          1
** errors        Ø
** warnings      Ø
                  31 *LLIST      ON
                  32 ;** SET UP NEW DOT COMMANDS **
FØØØ Ø634         4Ø init :LD    B,52 ;        INITIALISE THE 52 NEW DOT COMMANDS
FØØ2 219EF3              LD    HL,.A
FØØ5 3E34                LD    A,pno.routine-.A-1
FØØ7 77            loop :LD    (HL),A
FØØ8 3D                  DEC   A
FØØ9 23                  INC   HL
FØØA 1ØFB                DJNZ  loop
FØØC 21EFF3              LD    HL,no.routine
FØØF 22D3F3              LD    (pno.routine),HL
FØ12 3E1B         5Ø     LD    A,prem-REM-1 ; *NEW REM COMMAND*
FØ14 32BAF3              LD    (REM),A
FØ17 21C1F2              LD    HL,rem
FØ1A 22D6F3              LD    (prem),HL
FØ1D 3E17                LD    A,pif-IF-3 ;    *NEW IF COMMAND*
FØ1F 32CAF3              LD    (IF),A
FØ22 218BF3              LD    HL,if
FØ25 22DFF3              LD    (pif),HL
FØ28 3E14                LD    A,ppoke-POKE-2 ;*NEW POKE COMMAND*
FØ2A 32C4F3              LD    (POKE),A
FØ2D 21F1F3              LD    HL,poke
FØ3Ø 22DAF3              LD    (ppoke),HL
FØ33 3E43                LD    A,ptext-.A-6 ; *NEW DOT A COMMAND TO
FØ35 329EF3              LD    (.A),A ;          PRINT TEXT*
FØ38 21Ø8F4              LD    HL,print.text
FØ3B 22E7F3              LD    (ptext),HL
FØ3E FD365338            LD    (IY+83),56
FØ42 3E43                LD    A,pinput-.I-4 ;*NEW DOT I COMMAND TO
FØ44 32A6F3              LD    (.I),A ;          INPUT ON UPPER SCREEN*
FØ47 21CFF5              LD    HL,input
FØ4A 22EDF3              LD    (pinput),HL
                  54 *LLIST      OFF

** pass          2
** errors        Ø
** warnings      Ø
```

COMMAND	CODE-CEh	OFFSET ADDRESS	OFFSET VALUE	ADDRESS IN PARAM TABLE	CLASS CODES	ROUTINE ADDRESS
DEF FN	00	1A48	B1	1AF9	05	1F60
CAT	01	1A49	CB	1B14	0B	1793
FORMAT	02	1A4A	BC	1B06	0A 00	1793
MOVE	03	1A4B	BF	1B0A	0A 2C 0A 00	1793
ERASE	04	1A4C	C4	1B10	0A 00	1793
OPEN	05	1A4D	AF	1AFC	06 2C 0A 00	1736
CLOSE	06	1A4E	B4	1B02	06 00	16E5
MERGE	07	1A4F	93	1AE2	0B	
VERIFY	08	1A50	91	1AE1	0B	
BEEP	09	1A51	92	1AE3	08 00	03F8
CIRCLE	0A	1A52	95	1AE7	09 05	2320
INK	0B	1A53	98	1AEB	07	
PAPER	0C	1A54	98	1AEC	07	
FLASH	0D	1A55	98	1AED	07	
BRIGHT	0E	1A56	98	1AEE	07	
INVERSE	0F	1A57	98	1AEF	07	
OVER	10	1A58	98	1AF0	07	
OUT	11	1A59	98	1AF1	08 00	1E7A
LPRINT	12	1A5A	7F	1AD9	05	1FC9
LLIST	13	1A5B	81	1ADC	05	17F5
STOP	14	1A5C	2E	1A8A	00	1CEE
READ	15	1A5D	6C	1AC8	05	1DED
DATA	16	1A5E	6E	1ACC	05	1E27
RESTORE	17	1A5F	70	1ACF	03	1E42
NEW	18	1A60	48	1AA8	00	11B7
BORDER	19	1A61	94	1AF5	06 00	2294
CONTINUE	1A	1A62	56	1AB8	00	1E5F
DIM	1B	1A63	3F	1AA2	05	2C02
REM	1C	1A64	41	1AA5	05	1BB2
FOR	1D	1A65	2B	1A90	04 3D 06 CC 06 05	1D03
GOTO	1E	1A66	17	1A7D	06 00	1E67
GOSUB	1F	1A67	1F	1A86	06 00	1EED
INPUT	20	1A68	37	1A9F	05	2089
LOAD	21	1A69	77	1AE0	0B	
LIST	22	1A6A	44	1AAE	05	17F9
LET	23	1A6B	0F	1A7A	01 3D 02	
PAUSE	24	1A6C	59	1AC5	06 00	1F3A
NEXT	25	1A6D	2B	1A98	04 00	1DAB
POKE	26	1A6E	43	1AB1	08 00	1E80
PRINT	27	1A6F	2D	1A9C	05	1FCD
PLOT	28	1A70	51	1AC1	09 00	22DC
RUN	29	1A71	3A	1AAB	03	1EA1
SAVE	2A	1A72	6D	1ADF	0B	
RANDOMIZE	2B	1A73	42	1AB5	03	1E4F
IF	2C	1A74	0D	1A81	06 CB 05	1CF0
CLS	2D	1A75	49	1ABE	00	0D6B
DRAW	2E	1A76	5C	1AD2	09 05	2382
CLEAR	2F	1A77	44	1ABB	03	1EAC
RETURN	30	1A78	15	1A8D	00	1F23
COPY	31	1A79	5D	1AD6	00	0EAC

TABLE 1. COMMAND TO CLASS CODE CONVERSION

CLASS CODE	OFFSET ADDRESS	OFFSET VALUE	CLASS ROUTINE
00	1C01	0F	1C10
01	1C02	1D	1C1F
02	1C03	4B	1C4E
03	1C04	09	1C0D
04	1C05	67	1C6C
05	1C06	0B	1C11
06	1C07	7B	1C82
07	1C08	8E	1C96
08	1C09	71	1C7A
09	1C0A	B4	1CBE
0A	1C0B	81	1C8C
0B	1C0C	CF	1CDB

TABLE 2. CLASS CODE TO CLASS ROUTINE CONVERSION

00	NO EXPRESSION TO FOLLOW
01	'LET'
02	'LET'
03	EVALUATE NUMERIC EXPRESSION (DEFAULT ZERO)
04	EVALUATE SINGLE CHARACTER VARIABLE
05	EVALUATE 'A SET OF ITEMS'
06	EVALUATE NUMERIC EXPRESSION
07	'COLOUR COMMANDS'
08	EVALUATE TWO NUMERIC EXPRESSIONS (COMMA SEPARATOR)
09	HANDLE COLOURS THEN AS 08
0A	EVALUATE STRING EXPRESSION
0B	'TAPE ROUTINES'

NOTE: EVALUATION OF EXPRESSIONS TAKES PLACE DURING SYNTAX CHECKING AS WELL AS 'RUN TIME'. THE VALUE ENDS UP AS THE LAST 'VALUE' ON THE CALCULATOR STACK. IF THE EXPRESSION DOES NOT CONFORM TO THE REQUIRED TYPE THEN AN ERROR IS FLAGGED. IN SYNTAX CHECKS THIS RESULTS IN A RETURN TO THE EDITOR WITH A ? AT THE POINT OF ERROR.

TABLE 3. FUNCTION OF CLASS CODE ROUTINES

REMEMBER
CLASS ROUTINE 05 IS A *SPECIAL CASE* IN WHICH THE SYNTAX IS CHECKED WITHIN THE COMMAND ROUTINE AND MUST BE USED WITH CARE.

Stack

If we just take a little time out to briefly discuss 'values' on the calculator stack then perhaps the following will be a little clearer. The calculator stores two different types of value:

1. Numeric values of one form or another.
2. String parameters.

To get at these values there are several ROM routines that can prove extremely useful.

PROGRAM 1

Hexloader program.

```
   1 REM ****************
   2 REM ** HEXLOADER **
   3 REM ****************
   4 REM
   5 PAPER 7: INK 0: CLS : PRINT AT 7,7;"1. HEXLOADER": PRINT AT 9,7;"2. HEXDUMP
";AT 15,2;"Please select option 1 or 2"
   6 LET a$=INKEY$: IF a$<>"1" AND a$<>"2" THEN  GO TO 6
   7 IF a$="2" THEN  GO TO 1000
  10 DEF FN a()=INT (y/16)
  20 DEF FN b()=y-FN a()*16
  30 DEF FN c()=INT (x/256)
  40 DEF FN d()=x-FN c()*256
  50 DEF FN e()=INT (t/256)
  60 DEF FN f()=t-FN e()*256
  70 DEF FN g(a$,b)=CODE a$(b)-48-7*(CODE a$(b)>57)
  80 DEF FN h(a$)=16*FN g(a$,1)+FN g(a$,2)
  90 DEF FN c$()=CHR$ (FN a()+48+7*(FN a()>9))+CHR$ (FN b()+48+7*(FN b()>9))
 100 PAPER 7: INK 0: CLS
 110 LET r$="0"
 120 PRINT AT 11,7; FLASH 1;"REM SET CAPS LOCK";AT 13,3;"HAVE YOU CLEARED MEMORY
 ?"
 130 INPUT "Start address (DECIMAL) ";adr
 135 LET beg=adr
 140 PAPER 7: INK 0: CLS
 150 CLS : LET x=adr
 160 PRINT x
 180 INPUT "8 bytes + CHK. ";b$
 190 IF b$="" THEN  GO TO 180
 200 IF b$="ZZ" THEN  GO TO 450
 210 LET st=adr
 220 LET le=LEN b$: IF INT (le/2)*2<>le THEN  PRINT "Odd no. of chrs. ";: GO TO
410
 230 IF le=18 THEN  GO TO 250
 240 IF b$(le-1 TO )<>"ZZ" THEN  PRINT "No 'ZZ' marker ";: GO TO 410
 250 FOR a=1 TO le-2: IF b$(a)<"0" OR b$(a)>"F" THEN  PRINT "Invalid chr. ";: GO
 TO 410
 260 IF b$(a)<"A" AND b$(a)>"9" THEN  PRINT "Invalid chr. ";: GO TO 410
 270 NEXT a
 280 LET x=0
 290 FOR k=1 TO 8
 300 LET a$=b$( TO 2): LET b$=b$(3 TO )
 310 IF b$="ZZ" THEN  LET k=9: LET r$=b$: LET b$=a$: GO TO 360
 320 PRINT a$;" ";
 330 LET p=FN h(a$)
 340 LET x= x+p
 350 POKE adr,p: LET adr=adr+1
 360 NEXT k
 370 LET y=FN d(): PRINT "= ";FN c$();"/";b$
 380 LET a$=b$
 390 IF y=FN h(a$) THEN  GO TO 430
 400 PRINT "Checksum ";
 410 PRINT "error -re enter": BEEP .25,5: BEEP .5,2: LET adr=st
 420 GO TO 180
 430 BEEP .1,20
 440 IF r$<>"ZZ" THEN  GO TO 150
 450 PRINT "PROGRAM TERMINATED"
 460 PRINT "Address start = ";beg
 470 PRINT "Address   end = ";adr-1
 480 PRINT "Length of code = ";adr-beg
 490 STOP

 900 REM
 910 REM
 920 REM ************************
 930 REM ** HEXDUMP+CHECKSUM **
 940 REM ************************
 950 REM
1000 PAPER 7: INK 0: CLS : LET p=0
1010 INPUT "(S)creen OR (P)rinter ? ";q$
1020 IF q$="P" OR q$="p" THEN  LET p=1: GO TO 1040
1030 IF q$<>"S" AND q$<>"s" THEN  BEEP .5,2: GO TO 1010
1040 INPUT "From address (DECIMAL) ";st
1050 INPUT "To address (DECIMAL) ";end
1060 PAPER 7: INK 0: CLS
1070 FOR x=st TO end STEP 8
1080 LET y=FN c(): IF p THEN  LPRINT x;" ";: GO TO 1100
1085 PRINT x;" "
1100 LET t=0
1110 FOR z=0 TO 7
1120 IF x+z>end THEN  LET z=8: GO TO 1150
1130 LET y=PEEK (x+z): LET t=t+y
1140 IF p THEN  LPRINT FN c$();" ";: GO TO 1150
1145 PRINT FN c$();" ";
1150 NEXT z
1155 LET y=FN f()
1160 IF p THEN  LPRINT ;"= ";FN c$();: GO TO 1170
1165 PRINT "= ";FN c$();
1170 IF x+z>end THEN  GO TO 1210
1180 IF p THEN  LPRINT : GO TO 1200
1190 PRINT
1200 NEXT x
1210 IF p THEN  LPRINT "ZZ": STOP
1220 PRINT "ZZ"
```

1. STK.TO.A
Address = 2314h. 0-255 into A.
2. STK.TO.BC
Address = 2307h. Top value to B, next value to C.
3. TWO.PARAM
Address = 1E85h. Top value to A, next value to BC.
4. FIND.INT2
Address = 1E99h. Top value to BC.
5. STK. FETCH
Address = 2BF1h. DE = start of string, BC = length.

Now to get back to the POKE command 1E80h.

This is a very short program that uses 1E85h to take two numbers from the stack putting the first value (8) into the A register and the second value (65535) into the BC register pair. The instruction **LD (BC), A** is then used to **POKE 65535,8** and then a **Ret** is made to find the next statement, in our case there is no next statement so a return to the main execution loop is made via the report code 'OK' routine.

If, however, we had typed **POKE 65535,300** this would have passed syntax checking but would produce a run time, error report from the POKE command routine. The error would have been caused by trying to compress a number greater than 255 into the A register. Similarly, **POKE 655356,8** would give an error report as the BC register pair cannot hold numbers greater than 65535.

These errors are dealt with automatically by the RST 8 routine and need not concern us.

I include **Table 1** to give you all the commands available with their class codes and command routines where applicable.

From table 3 you will see that (leaving out 01, 02, 07 and 0B as 'dedicated classes') classes 00, 03, 04, 06, 08, 09 and 0A 'evaluate' expressions following the command and in fact do not call the command routine at all during syntax checking.

The exception to this is class 05 which is used in several commands, such as PRINT, READ, CIRCLE in which syntax is checked by calling the command routine following the class 05 code. In other words any expression following the command has to be checked for the syntax required within the command routine without actually performing the command.

For example, **PRINT "Hello";a$** has to be syntax checked by the PRINT command routine at IFCD without actually printing to the screen, and if you typed **PRINT "Hello";ab$** then the error must be spotted by the PRINT command routine, and flagged with a '?' after a.

New commands

Having got a grasp of the method of calculating a command class group we can dream up our own commands.

How about a FILL command that uses the format:

FILL colour; start co-ords x,y

eg FILL INK 2; PAPER 3; 127,88

[redacted]

this would have class codes 09 00 xxxx (xxxx being the FILL routine address)

Or, a LINE command which will draw a line from a point, x, y to a point a, b in any colour:

LINE INK 5; OVER 1; 100, 35; 150, 80

this would have class codes 09 3B 08 00 xxxx

The 3B code is included in the above CLASS CODES to indicate that a semicolon is required between the two sets of co ordinates. (We could of course have used any separator by placing its code between 09 and 08).

To achieve this we must set up an offset table and parameter class code table for our new commands and, what is more important, find a method of diverting the basic interpreter to consider using this new set of tables when encountering a new command.

Keywords

With the existing commands the Spectrum uses a single code/keyword combination so that command checking is made easier. For our extended Basic commands we will use a similar method, but to make it even easier we will procede a 'new command code' with the code 2Eh (FULL STOP). To save code space we will have single letter commands, any other method would entail having a routine to expand the new commands into keywords. In this extended Basic the upper and lower case letters A to Z will be used so that a new command will be in the form **.a** (Dot A) **.m** (Dot m) ie 'Dot commands'.

Now that we have the form of our new commands we can proceed to examine the way in which the ROM interpreter so that we can retain control of all the Spectrum commands as well as our own new ones. The routines required to do this are held in the code F08C to F39D. You can take my word for it that these routines work or disassemble them for yourself and using a suitable book on the Spectrum ROM, follow the routines chosen.

There is one small point to note with this method and this is to do with syntax checking of certain class code routines. Several Spectrum commands when checked for syntax will pass control back to the ROM interpreter *for the rest of the basic line only*.

This means that in some cases you cannot place new commands after normal commands as the syntax will fail. Should this occur then transfer the new command to another line. This does not affect new commands preceding normal ones.

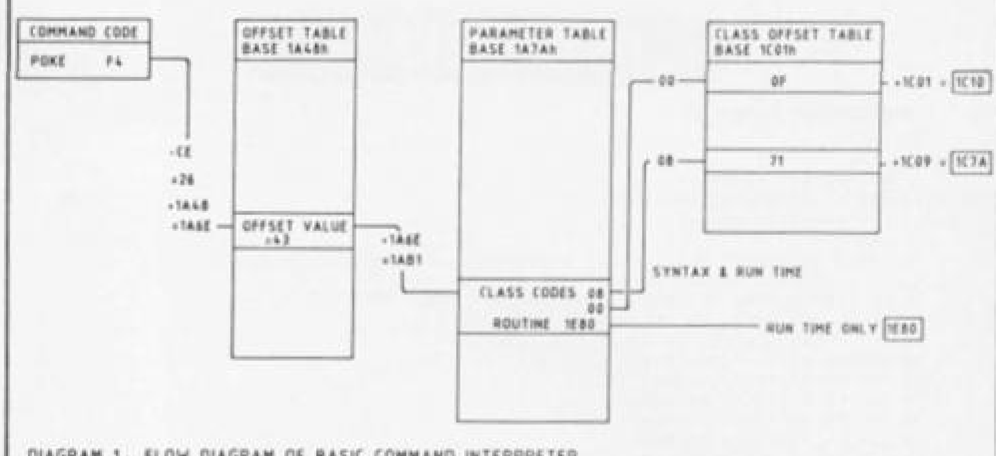


DIAGRAM 1. FLOW DIAGRAM OF BASIC COMMAND INTERPRETER

The reason for this problem is a little routine in the ROM called check end at 1BEEh, which is called many times when checking syntax to stop the execution of the actual command. This does not normally cause any problem but when syntax checking is carried out within the command routine as in class 05 then a call to this routine will cause a continuation into the statement. Next rom routine. In run time this call will have no effect as a direct return will be made back to the calling routine.

Dot commands

Now that we have control of the Basic interpreter we can modify it as we like to enable our dot commands to be recognised.

I include sections of the assembly listing to demonstrate how this is achieved.

LIST 1: Lines 250 to 350 show these modifications. In line 270 a statement beginning with a DOT command is diverted to use our new offset table whereas standard commands are handled in the normal manner using the ROM offset table.

You will notice that three normal commands REM, POKE and IF are also diverted to use our new offset table and are treated as dot commands:

REM = .c
POKE = .m
IF = .s

The REM and IF commands are special cases which if allowed to be handled by the ROM would cause control of the Basic interpreter to be lost. POKE has been diverted so that the command can be modified to error trap any POKES which would corrupt our interpreter.

Our new offet table is shown in my Assembly List 2 as LINE 600 and the Dot Command parameter table at LINE 650.

As an example of the method of setting up new commands I have included two new BASIC commands **.A** and **.I**

The **.A** command is a new text command and will print text on the screen in any size at any pixel position and in any colour (including INVERSE, BRIGHT and OVER etc. and if you look at the new parameter table the class codes for the command are:

09 3B 0A 3B 08 00

(this shows that you can string any number of class codes together to get the parameters you require).

An example of the command would be:

.A INK 0; PAPER 3; BRIGHT 1; 128,25; "HELLO";3,2

which will print 'HELLO' three characters by two characters wide) starting at pixel 128,25 using BLACK ink on MAGENTA paper, BRIGHT.

W=0 is a special case that will print at 42 characters per line and H=0 will only print the first seven bytes of the character form.

The **.I** command is an INPUT (line) command to the main screen and has class codes:

28 06 29 05

For example:

.I (5) INK 0; PAPER 5; FLASH 1; AT 3,5;"Name ? ";a$

will print an input prompt "Name ? " at line 3, column 5, wait for an input in the normal way and allocate the input (if the syntax is correct) to the string variable a$ The routine will allow inputs of codes 32 to 127 and DELETE. The maximum number of characters to be accepted is given by the numeric variable enclosed by the brackets immediately after the **.I**, in the above example this is '5'.

The **.A** routine occupies F408 to F51A and the **I.** command F51B to F72B.

Machine code hex data.

```
HEXDUMP DOT.COMMANDS
CODE 61440,1836

61440 06 34 21 9E F3 3E 34 77 = D5
61448 3D 23 10 FB 21 EF F3 22 = 90
61456 D3 F3 3E 1B 32 BA F3 21 = 1F
61464 C1 F2 22 D6 F3 3E 12 32 = 20
61472 CA F3 21 0B F3 22 DF F3 = 50
61480 3E 14 32 C4 F3 21 F1 F3 = 40
61488 22 DA F3 3E 43 32 9E F3 = 33
61496 21 08 F4 22 E7 F3 FD 36 = 4C
61504 53 38 3E 43 32 A6 F3 21 = F8
61512 CF F5 22 ED F3 06 07 21 = F4
61520 7E F0 7E 32 5B F0 23 C5 = 51
61528 E5 EF 34 EC 4C CC CC CC = A4
61536 38 E1 7E E5 CD 28 2D CD = 6B
61544 F8 03 E1 C1 23 10 E3 3E = F1
61552 01 CD 28 2D 3E 14 CD 28 = 6A
61560 2D CD F8 03 18 0E ED 0B = 13
61568 ED 0B F0 10 EC 0B EC 10 = EB
61576 EC 0B EC 08 2A B2 5C 36 = 5C
61584 3E 2B F9 11 00 F1 2B 2B = BA
61592 22 3D 5C D5 C3 85 F2 FD = C7
61600 36 31 02 CD 95 17 CD B0 = 5F
61608 16 3E 00 CD 01 16 CD 2C = 31
61616 0F CD EB F1 FD CB 00 7E = FE
61624 20 12 FD CB 30 66 28 40 = F8
61632 2A 59 5C CD A7 11 FD 36 = 97
61640 00 FF 18 DD 2A 59 5C 22 = F5
61648 5D 5C CD FB 19 78 B1 C2 = 85
61656 91 F1 DF FE 0D 28 C0 FD = 51
61664 CB 30 46 C4 AF 0D CD 6E = FC
61672 0D 3E 19 FD 96 4F 32 8C = 04
61680 5C FD CB 01 FE FD 36 00 = 56
61688 FF FD 36 0A 01 CD 99 F2 = 95
61696 76 FD CB 01 AE FD CB 30 = E5
61704 4E C4 CD 0E 3A 3A 5C 3C = F9
61712 F5 21 00 00 FD 74 37 FD = BB
61720 74 26 22 65 5C 21 01 00 = 9F
61728 22 16 5C CD B0 16 FD CB = EF
61736 37 AE CD 6E 0D FD CB 02 = F7
61744 EE F1 47 FE 0A 38 02 C6 = 2E
61752 07 CD EF 15 3E 20 D7 78 = 85
61760 11 91 13 CD 0A 0C AF 11 = 58
61768 36 15 CD 0A 0C ED 4B 45 = AB
61776 5C CD 1B 1A 3E 3A D7 ED = 9A
61784 4B 47 5C 06 00 CD 1B 1A = F6
61792 CD 97 10 3A 3A 5C 3C 28 = A8
61800 1D FE 09 28 04 FE 15 20 = 83
61808 03 FD 34 0D 01 03 00 11 = 56
61816 70 5C 21 44 5C FD CB 0A = 5F
61824 7E 28 01 09 ED B8 FD 36 = 88
61832 0A FF FD CB 01 9E C3 A9 = DC
61840 F0 ED 43 49 5C 2A 5D 5C = A8
61848 EB 21 E3 F1 E5 2A 61 5C = AC
61856 37 ED 52 E5 60 69 CD 6E = 5F
61864 19 20 06 CD B8 19 CD E8 = 92
61872 19 C1 79 3D B0 28 28 C5 = 55
61880 03 03 03 03 2B ED 5B 53 = D2
61888 5C D5 CD 55 16 E1 22 53 = BF
61896 5C C1 C5 13 2A 61 5C 2B = 07
61904 2B ED B8 2A 49 5C EB C1 = 4B
61912 70 2B 71 2B 73 2B 72 F1 = 38
61920 C3 9F F0 3E 10 01 00 00 = A1
61928 C3 10 F1 FD CB 01 BE CD = 18
61936 FB 19 AF 32 47 5C 3D 32 = 07
61944 3A 5C 18 01 E7 CD BF 16 = 38
61952 FD 34 0D FA 8A 1C DF 06 = C3
61960 00 FE 0D CA C2 F2 FE 3A = C1
61968 28 EA 21 85 F2 E5 4F E7 = C5
61976 21 9E F3 79 FE 2E 28 1A = 99
61984 FE F4 28 08 FE FA 28 07 = 4C
61992 FE EA 28 03 21 48 1A D6 = 6C
62000 CE DA 8A 1C 4F 09 4E 09 = FD
62008 18 23 DF 4F E7 79 CB AF = 43
62016 FE 5B D2 8A 1C D6 41 DA = C2
62024 8A 1C 79 FE 61 38 02 D6 = 8E
62032 06 D6 41 21 9E F3 06 00 = D5
62040 18 DA 2A 74 5C 7E 23 22 = AF
62048 74 5C 01 5A F2 C5 21 01 = 04
62056 1C 4F FE 20 30 10 FE 06 = CD
62064 30 03 21 10 F3 06 00 09 = 66
62072 4E 09 E5 DF 05 C9 DF B9 = 81
62080 C2 8A 1C E7 C9 CD 54 1F = 58
62088 38 02 CF 14 FD CB 0A 7E = 6D
62096 20 71 2A 42 5C CB 7C 28 = C8
62104 14 21 FE FF 22 45 5C 2A = 1F
62112 61 5C 2B ED 5B 59 5C 1B = 00
62120 3A 44 5C 18 33 CD 6E 19 = 79
62128 3A 44 5C 28 19 A7 20 43 = 25
62136 47 7E E6 C0 78 28 0F CF = E9
62144 FF C1 CD 30 25 C8 2A 55 = 29
62152 5C 3E C0 A6 C0 AF FE 01 = 6E
62160 CE 00 56 23 5E ED 53 45 = 2A
62168 5C 23 5E 23 56 EB 19 23 = 7D
62176 22 55 5C EB 22 5D 5C 57 = F0
62184 1E 00 FD 36 0A FF 15 FD = 6C
62192 72 0D CA FC F1 14 CD 8B = A2
62200 19 28 08 CF 16 CD 30 25 = 50
62208 C0 C1 C1 DF FE 0D 28 BA = 0E
62216 FE 3A CA FC F1 C3 8A 1C = 58
62224 0F 1D 4B 09 67 0B 00 00 = F2
62232 00 00 00 00 CD DE 1C BF = 86
62240 C1 CC FD F2 EB 2A 74 5C = 61
62248 4E 23 46 EB C5 C9 CD B2 = AF
62256 28 FD 36 37 00 30 08 FD = C7
62264 CB 37 CE 20 18 CF 01 CC = A4
62272 96 29 FD CB 01 76 20 0D = 2B
62280 AF CD 30 25 C4 F1 2B 21 = D2
62288 71 5C B6 77 EB ED 43 72 = 87
62296 5C 22 4D 5C C9 C1 CD 65 = E3
62304 F3 CD FD F2 C9 3A 3B 5C = 49
62312 F5 CD FB 24 F1 FD 56 01 = 26
62320 AA E6 40 20 13 CB 7A C2 = 0A
62328 FF 2A C9 CD B2 28 F5 79 = 07
62336 F6 9F 3C 20 03 F1 18 A9 = A6
62344 C3 8A 1C C1 CD 30 25 28 = 74
62352 0A EF 02 38 EB CD E9 34 = 08
62360 DA C2 F2 C3 FD F1 43 33 = B5
62368 32 31 30 2F 2E 2D 43 2B = 8B
62376 2A 29 28 27 26 25 24 23 = 34
62384 22 21 20 1F 1E 1D 1C 1B = F4
62392 1A 19 18 17 16 15 14 13 = B7
62400 12 11 10 0F 14 0D 0C 0B = 7A
62408 0A 09 12 07 06 05 04 03 = 3E
62416 02 01 00 EF F3 05 C1 F2 = 9D
62424 08 00 F1 F3 06 CB 05 8B = 4D
62432 F3 09 3B 0A 3B 08 00 08 = BC
62440 F4 28 06 29 05 CF F5 CF = E3
62448 0B CD 85 1E 21 4F FF A7 = 91
62456 ED 42 38 0A 21 FF EF A7 = 27
62464 ED 42 30 02 CF 19 02 C9 = 14
62472 CD 07 23 21 6B F4 36 08 = B5
62480 78 FE 21 D2 F9 24 A7 20 = 4D
62488 03 35 35 04 21 6D F4 36 = 29
62496 08 79 FE 19 D2 F9 24 A7 = 2E
62504 20 02 35 0C ED 43 70 F4 = F7
62512 CD F1 2B 78 B1 C8 ED 53 = 1A
62520 66 F4 ED 43 68 F4 CD 07 = 8A
62528 23 78 FE C0 D2 F9 24 ED = 35
62536 43 6E F4 ED 43 72 F4 ED = 28
62544 4B 68 F4 ED 5B 66 F4 1A = 63
62552 C5 D5 CD 74 F4 D1 C1 13 = 74
62560 0B 78 B1 20 F2 C9 0C 70 = 8B
62568 25 00 00 06 00 07 E3 74 = 89
62576 01 01 E3 74 26 00 6F 29 = 17
62584 29 29 ED 5B 36 5C 19 ED = 32
62592 4B 6C F4 CB 58 20 01 23 = 12
62600 C5 ED 4B 6F F4 E5 C5 ED = F7
62608 4B 6A F4 CB 58 7E 20 01 = 6B
62616 17 C5 17 F5 38 1B ED 4B = 73
62624 70 F4 C5 ED 4B 72 F4 C5 = 8C
62632 CD 10 F5 CD DB 0B C1 0C = 52
62640 ED 43 72 F4 C1 10 EB 18 = 6A
62648 16 ED 4B 70 F4 C5 ED 4B = AF
62656 72 F4 C5 CD 0A F5 C1 0C = C4
62664 ED 43 72 F4 C1 10 EE F1 = 46
62672 C1 10 C6 3A 6E F4 21 72 = C6
62680 F4 77 23 7E A7 20 02 3E = 13
62688 C0 3D 77 C1 E1 10 A6 23 = EF
62696 C1 10 9D 3A 6B F4 FE 06 = 0B
62704 28 06 3A 71 F4 87 87 87 = 62
62712 6F 3A 72 F4 85 32 6E F4 = 28
62720 32 72 F4 3A 6F F4 32 73 = DA
62728 F4 C9 CD 10 F5 C3 EC 22 = 60
62736 3E BF 90 30 02 06 BF C3 = 47
62744 B0 22 C9 CD E0 F6 FD CB = 06
62752 01 6E 28 FA 3A 08 5C FD = 2C
62760 CB 01 AE FE 0D C8 FE 0C = 57
62768 C8 FE 06 20 14 3A 6A 5C = 00
62776 EE 08 32 6A 5C 3A 08 F6 = 26
62784 A7 28 D8 21 06 F6 34 18 = 10
62792 D2 FE 20 38 D1 FE 80 30 = A7
62800 CD C9 2A 3D 5C E5 21 BD = 1C
62808 F5 E5 22 3D 5C CD 1B F5 = 72
62816 F5 16 00 FD 5E 01 21 C8 = 50
62824 00 CD B5 03 F1 21 5D F5 = E9
62832 E5 FE 0D 28 3B FE 0C 28 = 85
62840 1D 6F 3A 07 F6 A7 C8 3D = 6F
62848 32 07 F6 7D 2A 5B 5C CD = 5A
62856 57 16 12 13 ED 53 5B 5C = 84
62864 21 06 F6 36 00 C9 2A 5B = A1
62872 5C CD 31 10 01 01 00 CD = 39
62880 E8 19 21 06 F6 34 3A 08 = 94
62888 F6 A7 28 01 34 23 34 C9 = 1A
62896 E1 E1 E1 22 3D 5C FD CB = 26
62904 00 7E C0 F9 C9 FD 36 00 = 33
62912 FF 16 00 FD 5E 02 21 90 = 23
62920 1A CD B5 03 C3 56 F5 CD = 7A
62928 30 25 28 05 3E 02 CD 01 = 90
62936 16 CD E0 F5 CD FD F2 C9 = 3D
62944 CD 30 25 28 0A CD 14 23 = 58
62952 32 07 F6 A7 CA F9 24 CD = 8A
62960 4E 20 28 FB FE 28 20 11 = E8
62968 E7 CD DF 1F DF FE 29 C2 = 7A
62976 8A 1C E7 C3 D9 F6 00 01 = 20
62984 00 FE CA 20 11 E7 CD 1F = CC
62992 1C FD CB 37 FE FD CB 01 = E2
63000 76 C2 8A 1C 18 0D CD 8D = 5D
63008 2C D2 D6 F6 CD 1F 1C FD = CF
63016 CB 37 BE CD 30 25 CA D9 = 85
63024 F6 CD BF 16 21 71 5C CB = 51
63032 B6 CB EE 01 01 00 CB 7E = BA
63040 20 0B 3A 3B 5C E6 40 20 = 42
63048 02 0E 03 B6 77 F7 36 0D = 7A
63056 79 0F 0F 30 05 3E 22 12 = 3E
63064 2B 77 22 5B 5C FD CB 37 = 7A
63072 7E 20 26 2A 5D 5C E5 2A = B6
63080 3D 5C E5 21 6B F6 E5 ED = D2
63088 73 3D 5C 2A 61 5C CD A7 = 67
63096 11 FD 36 00 FF CD 52 F5 = 57
63104 FD CB 01 BE CD B9 21 18 = 46
63112 03 CD 52 F5 FD 36 22 00 = 6C
63120 37 CD 95 11 EB CD 7D 18 = F7
63128 3E 20 D7 21 71 5C CB AE = 9C
63136 CB 7E CB BE 20 1C E1 E1 = D0
63144 22 3D 5C E1 22 5F 5C FD = 76
63152 CB 01 FE CD B9 21 2A 5F = FA
63160 5C FD 36 26 00 22 5D 5C = 90
63168 18 17 2A 63 5C ED 5B 61 = C1
63176 5C 37 ED 52 44 4D CD B2 = E2
63184 2A CD FF 2A 18 03 CD FC = 04
63192 1F CD 4E 20 CA EF F5 C9 = D1
63200 FD 36 52 FF CD 4D 0D 2A = D5
63208 84 5C E5 2A 88 5C E5 37 = EF
63216 CD 95 11 EB CD 7D 18 EB = AB
63224 CD E1 18 3A 06 F6 A7 28 = CB
63232 09 3D 32 06 F6 3E 20 D7 = A9
63240 18 F1 C1 E1 11 20 00 3A = 16
63248 8C 5C 3C 28 05 04 ED 52 = 94
63256 18 F8 ED 43 88 5C 22 84 = CA
63264 5C FD 7E 26 32 08 F6 FD = 2A
63272 36 26 00 C9 = 2522
```

Offset values

Finally, the routine in **assembly list 3** was used to set up the correct offset table values for all the dot commands. I included this because it saved calculating values every time a change was made to the command routines. The Laser Genius assembler allows mathematical calculations based on the label address values as long as the result is within the range of the machine code command, ie LD A,prem — REM — 1 means;

LD A,(address prem) — (address REM) — 1

and must be a value less than 256.

The code has a start address of F000h (61440d) and should be called from this address, a fanfare BEEP will be played to indicate the routine has been installed.

My dual purpose **HEXLOADER/ HEXDUMP** program **(program 1)** should be used to enter the code as list 4. Once entered it should be saved as **"DOTCODE" CODE 61440,1836.**

After the CODE has been saved it can be loaded at the start of your extended Basic program such that it will AUTO RUN using something along the lines of the example in **program 2:**

PROGRAM 2

Example Basic program.

```
10 REM ** START OF EXTENDED
   BASIC PROGRAM **

20
.
.
.
ETC.
.
.
.
.
9000 STOP
9900 CLEAR 61439
9910 LOAD "DOTCODE" CODE 61440
9920 RANDOMIZE USR 61440: GOTO 10
9930 SAVE "BASICPROG" LINE 9900
9940 SAVE "DOTCODE" CODE 61440,1836
9950 PRINT "REWIND TAPE TO VERIFY . ."
9960 VERIFY " " :VERIFY " " CODE
9970 PRINT "VERIFY O.K. . . ."
```

Should you wish to exit extended BASIC use the NEW command which will revert back to the ROM interpreter.

And that as they is all there is to it. I hope that you have got this far with some understanding of the methods discussed and that you now feel able to tackle one of your own new commands, how about the LINE command touched upon earlier?

Alan Davis presents a high quality plotting utility

SUPA-PLOT

Despite what the cynics say, we all know that the Spectrum is very much more than "just a games machine". Many of us use our machines for a great variety of purposes: perhaps for managing business accounts, collating statistics, or for scientific work — to name but a few. For example, in between adventure games my own Spectrum has performed more statistical calculations in the last few years alone than I could otherwise have done myself (even with the aid of a pocket calculator) in a lifetime. That, as they say, is the good news.

The bad news? Well, have you ever tried to put the graphics capabilities of the Spectrum to serious use for detailed, accurate illustrations on paper? The usual method, of course, is to PLOT, PRINT, etc. onto the screen, and COPY the result. This way, you get adequate resolution for block histograms or bar charts of some sort, where precision plotting isn't essential. But suppose you want to plot a graph of the measurements from that experiment you did at school last Thursday? Or how about a statistical scatter diagram? Or survey points for a site plan? How many times have you reached for the graph paper and pencil to tackle this sort of job simply because a screen dump from the Spectrum would obviously be too crude and inaccurate?

If your head is nodding in sympathy, then help is now at hand in the shape of SUPA-PLOT. But before you get carried away and burn all your graph paper, I'd better give you a few essential details straight away. SUPA-PLOT has been written to work with the Amstrad DMP 2000 dot matrix printer. This uses standard EPSON control codes — and so long as your printer is Epson compatible then all should be well. However, in addition to this, your printer MUST be capable of the following:

(a) 1/216" line feeds, in both forward and reverse directions.
(b) Quadruple density graphics mode.

If your printer lacks any of these features, then I'm afraid you're still stuck with the graph paper and pencil. Sorry.

The basic idea behind SUPA-PLOT is to ignore the Spectrum screen altogether and drive the printer directly, under software control. The reason for this is simple — the printer itself is capable of terrific resolution both along a line and in the paper feed direction. I decided at the outset that a printing area of 5" by 5" would be suitable for most purposes, so let's see what this means in terms of resolution. In the "x" direction (i.e. along a horizontal line of printing), the spacing of printable dots in quadruple density graphics mode is 1/240". So by putting the printer into quad graphics mode and then sending a suitable number of zero codes (CHR$ 0), a printing position can be defined with a precision of 1/240". Along our chosen 5" scale, then, we'll get 1200 possible print positions — over four times as many as the Spectrum screen display would permit, with each position defined MORE precisely than you could achieve with pencil and graph paper of comparable size!

What about the "y" direction, at 90 degrees to the print lines? In principle, by using control codes for forward and reverse line feeds, the paper can be shunted back and forth to define a position with a precision of 1/216". Not quite so good as the "x" direction, then — but still more than acceptable, and giving a total of 1080 print positions along our vertical 5" scale.

I say "in principle" advisedly, because when I began writing SUPA-PLOT, I was by no means certain that the end result would be good enough. It was clear that a very great deal of paper feeding (both backwards and forwards) would be involved during a typical plotting session, and I had no idea whether the mechanics of the printer were up to the job. I needn't have worried. The performance of the Amstrad confirmed my opinion that it's a magnificent piece of precision engineering. I don't know how the designers managed to eliminate the effects of backlash in the paper feeds, but they seem to have done it — and I'll be interested to have some feedback about how well other machines perform with SUPA-PLOT. Perhaps you could let me know?

However, it should be obvious from what we've seen so far that in theory, at least, we can drive the printer as an ultra-high resolution plotter. Any point with co-ordinates (x,y) can be plotted by first feeding the paper an appropriate number of 1/216" line feeds ("y") from a chosen fixed position, and then printing a suitable number of zero codes in quad graphics mode ("x") before plotting the point. A variable scaling factor is needed for each axis of course so that different ranges of x and y values can be dealt with, but that's easily done.

Basic

In a situation like this, BASIC is perfectly adequate for most of the work (since the bulk of the time taken to complete a given plot is determined by the physical speed of the printer itself) — but there's one exception: shoving out the "zero" codes to the printer to define the "x" position. In BASIC this would involve a FOR/NEXT loop of up to 1200 cycles, and this particular task is obviously going to be best done in machine code. Listing 1 gives the assembler program we need.

Listing 1

```
*HISOFT GENS3M2 ASSEMBLER*
       ZX SPECTRUM

Copyright (C) HISOFT 1983,4
All rights reserved

Pass 1 errors: 00

         10 *C-
         20 *D+
         30 ; Routine for sending repeated printer codes
         40
65368    50          ORG  65368; UDG area
65368    60          LD   A,3
65370    70          CALL #1601; Select printer for output
65373    80          LD   BC,(COUNT); BC is the counter
65377    90 LOOP     LD   A,B
65378   100          OR   C
65379   110          RET  Z; Return to BASIC if count complete
65380   120          LD   A,(CODE)
65383   130          RST  16; Send the printer code
65384   140          DEC  BC; Decrease the counter
65385   150          JP   LOOP; Back for another
65388   160 CODE     DEFB 0; POKE this with the code to be sent
65389   170 COUNT    DEFW 00; POKE these two bytes with the count

Pass 2 errors: 00

Table used:    47  from   156
```

The code is assembled in the first few bytes of the UDG area, and to use it we need to POKE the address labelled COUNT (65389/90) with the number of codes to be sent (treating it as a 2-byte number), POKE the address CODE (65388) with the code to be sent (usually zero) — and then do a **USR 65368** call.

From this point, the programming task should have been very straightforward — just a case of linking the appro priate control codes together and keeping track of the line feeds so that at all times the program can return the print position to a fixed reference point on the paper. Well, maybe it's just me, but I can't recall any other short routine that has caused me so much trouble to write. The main problem was that testing it proved a messy and time-consuming job, particularly since bugs are likely to send the printer into overdrive, spewing out reams of paper and spattering random symbols everywhere! By the time I was at last able to shout "Eureka!", the room was knee-deep in wasted tractor paper . . .

Was it worth it? My word, it most certainly was — so much so that I decided to build the plotting routine into a self-contained utility with compre-

Listing 2

```
   1 DEF FN h(x)=INT (x/256): DE
F FN l(x)=x-256*FN h(x)
   7 REM ******************
   8 REM **COLLECT INPUT**
   9 REM ******************
  10 INPUT AT 0,0;(a$);" (or ENT
ER)"' LINE z$: IF LEN z$>16 THEN
  GO TO 10
  11 IF z$="" THEN  LET val=0: R
ETURN
  12 FOR i=1 TO LEN z$: IF (z$(i
)<"0" OR z$(i)>"9") AND z$(i)<>"
." THEN  LET val=0: GO TO 15
  13 NEXT i
  14 LET val=ABS VAL z$
  15 RETURN
  17 REM *************************
  18 REM *QUAD DENSITY GRAPHICS*
  19 REM *************************
  20 LPRINT e$;"Z";CHR$ FN l(cou
nt);CHR$ FN h(count);
  21 RETURN
  27 REM *************************
  28 REM **SEND REPEATED CODES**
  29 REM *************************
  30 POKE 65389,FN l(count): POK
E 65390,FN h(count)
  31 POKE 65388,code: LET m=USR
65368: RETURN
  37 REM ******************
  38 REM **REVERSE FEEDS**
  39 REM ******************
  40 FOR j=1 TO y2: LPRINT e$;"j
";CHR$ 255;: NEXT j: LPRINT e$;"
j";CHR$ y1;: RETURN
  47 REM ******************
  48 REM **FORWARD FEEDS**
  49 REM ******************
  50 FOR j=1 TO y2: LPRINT e$;"J
";CHR$ 255;: NEXT j: LPRINT e$;"
J";CHR$ y1: RETURN
  57 REM ******************
  58 REM *SINGLE KEYPRESS*
  59 REM ******************
  60 PAUSE 0: LET i$=INKEY$: IF
i$<CHR$ 123 AND i$>CHR$ 96 THEN
 LET i$=CHR$ (CODE i$-32)
  65 RETURN
 997 REM **************
 998 REM **MAIN MENU**
 999 REM **************
1000 CLS : PRINT AT 1,7;"HI-RES
PLOTTER"
1005 PRINT AT 3,7; INVERSE 1;"Se
lect:"
1010 PRINT AT 5,4;"1: Define axe
s";AT 7,4;"2: Test display";AT 9
,4;"3: Hard copy"
1015 PRINT AT 11,4;"4: Edit data
";AT 13,4;"5: Save data";AT 15,4
;"6: Load data";AT 20,4;"Q: Fres
h start"
1020 GO SUB 60: IF i$="Q" THEN
GO TO 8000
1030 IF i$<"1" OR i$>"6" THEN  G
O TO 1020
1035 GO SUB 4500+500*VAL i$: GO
TO 1000
4997 REM ****************
4998 REM **DEFINE AXES**
4999 REM ****************
5000 CLS : PRINT  INVERSE 1;AT 0
,13;"TITLE"; INVERSE 0;AT 1,0;t$
5005 PRINT AT 4,0; INVERSE 1;"LA
BELS"; INVERSE 0;" Y: ";y$;AT 5,
7;"X: ";x$
5010 PRINT AT 7,0; INVERSE 1;"Y
AXIS"; INVERSE 0;" Range ";ymin;
" - ";ymax;AT 8,1;"(";ydots/216;
"""")";AT 8,7;"Scale interval ";y
int
5015 PRINT AT 10,0; INVERSE 1;"X
 AXIS"; INVERSE 0;" Range ";xmin
;" - ";xmax;AT 11,1;"(";xdots/24
0;"""")";AT 11,7;"Scale interval
";xint
5020 PRINT AT 14,7; INVERSE 1;"S
ELECT:"; INVERSE 0;AT 16,7;"Y to
 change Y axis";AT 17,7;"X to ch
ange X axis";AT 18,7;"T to chang
e text";AT 19,7;"F to change for
mat";AT 20,7;"ENTER to quit"
5025 GO SUB 60
5030 IF i$=CHR$ 13 THEN  GO TO 5
100
5035 IF i$="X" THEN  PRINT AT 10
,0; FLASH 1;"X AXIS": LET a$="Mi
nimum X value?": GO SUB 10: LET
xmin=val+(xmin AND z$=""): PRINT
 AT 10,13,,AT 10,13;xmin;" - ";:
 LET a$="Maximum X value?": GO S
UB 10: LET xmax=val+(xmax AND z$
=""): PRINT xmax,: LET a$="Scale
 interval?": GO SUB 10: LET xint
=val+(xint AND z$=""): PRINT AT
11,22;xint,AT 10,0; INVERSE 1;"X
 AXIS"
5040 IF i$="Y" THEN  PRINT AT 7,
0; FLASH 1;"Y AXIS": LET a$="Min
imum Y value?": GO SUB 10: LET y
min=val+(ymin AND z$=""): PRINT
AT 7,13,,AT 7,13;ymin;" - ";: LE
T a$="Maximum Y value?": GO SUB
10: LET ymax=val+(ymax AND z$=""
): PRINT ymax,: LET a$="Scale in
terval?": GO SUB 10: LET yint=va
l+(yint AND z$=""): PRINT AT 8,2
2;yint,AT 7,0; INVERSE 1;"Y AXIS
"
5045 IF i$="T" THEN  PRINT AT 4,
0; FLASH 1;"LABELS": LET a$="Y-a
xis label?": GO SUB 10: LET y$=z
$+(y$ AND z$=""): PRINT AT 4,10,
,AT 4,10;y$: LET a$="X-axis labe
l?": GO SUB 10: LET x$=z$+(x$ AN
D z$=""): PRINT AT 5,10,,AT 5,10
;x$: INPUT AT 0,0;"Main title?"'
 LINE z$: LET t$=z$+(t$ AND z$="
"): PRINT AT 1,0,,,,AT 1,0;t$;AT
 4,0; INVERSE 1;"LABELS"
5050 IF i$="F" THEN  LET a$="Len
gth of Y axis ("")?": GO SUB 10:
 LET val=INT (val+.5) AND val<=8
: LET ydots=(ydots AND NOT val)+
216*val: PRINT AT 8,1;"(";ydots/
216;"""")": LET a$="Length of X a
xis("")?": GO SUB 10: LET val=IN
T (val+.5) AND val<=5: LET xdots
=(xdots AND NOT val)+240*val: PR
INT AT 11,1;"(";xdots/240;"""")"
5060 GO TO 5025
5100 LET scx=xdots/(xmax-xmin):
LET scy=ydots/(ymax-ymin): RETUR
N
5497 REM ******************
5498 REM **TEST DISPLAY**
5499 REM ******************
5500 CLS : PRINT AT 3,20;"TESTIN
G";AT 4,20;"DISPLAY": GO SUB 510
0
5510 LET st=87-ydots/20: PLOT 0,
st: DRAW xdots/10,0: DRAW 0,ydot
s/10: DRAW -xdots/10,0: DRAW 0,-
ydots/10
5515 FOR i=1 TO c(max,3): IF c(i
,1)>xmax OR c(i,1)<xmin OR c(i,2
)>ymax OR c(i,2)<ymin THEN  PRIN
T AT 15,20; FLASH 1;"WARNING";AT
 17,16;"LIMITS EXCEEDED": GO TO
5525
5520 PLOT INT (.5+(c(i,1)-xmin)*
scx/10),st+INT (.5+(c(i,2)-ymin)
*scy/10)
5525 NEXT i
5549 PRINT #1;AT 1,10;"PRESS A K
EY": PAUSE 0
5550 RETURN
5997 REM **********************
5998 REM **HI-RES HARD COPY**
5999 REM **********************
6000 REM *** "TOKENS OFF" ***
6005 COPY : REM CHR$ 0
6010 CLS : PRINT AT 10,9;"PLOTTI
NG..."
6014 REM ***RESET PRINTER***
6015 LPRINT e$;"@"
6019 REM **AUTO LF OFF/MARGIN**
6020 LPRINT e$;"A";CHR$ 0;e$;"l"
;CHR$ 20
6024 REM ***DRAW Y AXIS***
6025 FOR i=1 TO ydots/24
6030 LET count=2: GO SUB 20: LPR
INT CHR$ 255;CHR$ 255
6033 LPRINT e$;"J";CHR$ 1;
6036 GO SUB 20: LPRINT CHR$ 255;
CHR$ 255
6037 LPRINT e$;"J";CHR$ 23;
6040 NEXT i: LPRINT e$;"j";CHR$
24
6049 REM ***DRAW X AXIS***
6050 LET count=xdots: GO SUB 20
6060 LET code=1: GO SUB 30
6064 REM ***Y AXIS DIVISIONS***
6065 LPRINT e$;"l";CHR$ 19
6070 FOR i=0 TO ydots/(scy*yint)
6075 LET count=INT (.5+i*scy*yin
t): LET y1=FN l(count): LET y2=F
N h(count): GO SUB 40
6080 LET count=12: GO SUB 20: LE
T code=0: GO SUB 30: GO SUB 20:
LET code=1: GO SUB 30: GO SUB 50
6085 NEXT i: LPRINT e$;"J";CHR$
24;e$;"l";CHR$ 20
6089 REM ***X AXIS DIVISIONS***
6090 FOR i=0 TO xdots/(scx*xint)
6095 LET count=2+INT (.5+i*scx*x
int)
6100 GO SUB 20: LET count=count-
2: LET code=0: GO SUB 30: LPRINT
 CHR$ 248;CHR$ 248
6105 NEXT i: LPRINT e$;"l";CHR$
19
6109 REM ***PLOT THE POINTS***
6110 FOR i=1 TO c(max,3)
6120 LET x=32+INT (.5+(c(i,1)-xm
in)*scx): LET y=INT (.5+(c(i,2)-
ymin)*scy)+12
6130 LET y2=FN h(y): LET y1=FN l
(y)
```

```
6140 GO SUB 40
6150 LET count=x: GO SUB 20
6160 LET code=0: LET count=x-18:
 GO SUB 30
6170 RESTORE 9000+(c(i,3) AND c(
i,3)<4): FOR j=1 TO 18: READ g:
LPRINT CHR$ g;: NEXT j
6180 GO SUB 50
6190 NEXT i
6199 REM ***LABELS & TEXT***
6200 LET len1=LEN STR$ ymin: LET
 len2=LEN STR$ ymax
6205 LPRINT e$;"G";e$;"E";
6210 LPRINT e$;"j";CHR$ 12;e$;"l
";CHR$ (19-len1)
6215 LPRINT STR$ ymin: LPRINT e$
;"l";CHR$ (19-len2)
6220 LET y2=FN h(ydots): LET y1=
FN l(ydots): GO SUB 40
6230 LPRINT STR$ ymax: GO SUB 50
6235 LET centre=INT ((24+ydots)/
2): LET y2=FN h(centre): LET y1=
FN l(centre): GO SUB 40
6240 LPRINT e$;"l";CHR$ (INT (.5
*(19-LEN y$)))
6245 LPRINT y$: GO SUB 50
6250 LET len1=LEN STR$ xmin: LET
 len2=LEN STR$ xmax
6255 LPRINT e$;"J";CHR$ 36;e$;"l
";CHR$ (19-(INT (len1/2))): LET
count=12: GO SUB 20: LET code=0:
 GO SUB 30
6260 LPRINT STR$ xmin
6265 LPRINT e$;"l";CHR$ 20
6270 LET count=xdots-12-24*INT (
len2/2)
6275 GO SUB 20
6280 LET code=0: GO SUB 30: LPRI
NT STR$ xmax
6285 LPRINT e$;"J";CHR$ 48
6290 LET count=INT (xdots/2-12-2
4*LEN x$/2): GO SUB 20
6295 LET code=0: GO SUB 30: LPRI
NT x$
6300 LPRINT e$;"J";CHR$ 96;t$
6310 LPRINT CHR$ 12
6400 RETURN
6497 REM **************
6498 REM **EDIT DATA**
6499 REM **************
6500 LET auto=1: LET n=c(max,3):
 LET page=0: LET pos=1
6510 CLS : PRINT  INVERSE 1;AT 0
,3;"  X  ";AT 0,13;"  Y  ";AT 0,
25;"MARKER"
6520 LET limit=20*page+20: FOR i
=1+20*page TO (limit AND limit<n
)+(n AND limit>=n)
6530 PRINT  OVER 1;TAB 6-LEN STR
$ INT c(i,1);c(i,1);TAB 16-LEN S
TR$ INT c(i,2);c(i,2);TAB 27;c(i
,3);TAB 29;CHR$ (150+c(i,3))
6540 NEXT i
6545 PRINT  INVERSE 1;AT 21,0;"A
UTO (A)" AND auto;AT 21,12;"PAGE
 ";page+1: PRINT #1;AT 0,0; PAPE
R 1;"6/7=MOVE CURSOR : 5/8=NEXT
PAGE X/Y/M=EDIT : CS/0=DELETE EN
TRY  "
6550 LET count=pos+20*page: PRIN
T AT pos,0; OVER 1; PAPER 2,,
6560 GO SUB 60: IF i$=CHR$ 13 TH
EN  RETURN
6570 IF i$="6" OR i$="7" THEN  P
RINT AT pos,0; OVER 1; PAPER 0,,
: LET pos=pos+(i$="6" AND pos<20
 AND count<=n AND count<max-1)-(
i$="7" AND pos>1): GO TO 6550
6580 IF i$="5" AND page THEN  LE
T pos=1: LET page=page-1: GO TO
6510
6590 IF i$="8" AND 20*(page+1)<=
n AND count<max-1 THEN  LET pos=
1: LET page=page+1: GO TO 6510
6600 IF i$="X" THEN  LET a$="New
 X value?": GO SUB 10: LET c(cou
nt,1)=val: GO SUB 6700
6610 IF i$="Y" THEN  LET a$="New
 Y value?": GO SUB 10: LET c(cou
nt,2)=val: GO SUB 6700
6620 IF i$="M" THEN  LET a$="New
 marker number?": GO SUB 10: LET
 c(count,3)=val AND val<4: GO SU
B 6700
6630 IF i$=CHR$ 12 AND n AND cou
nt<=n THEN  PRINT #1;AT 0,0,,,,;
AT 0,11; FLASH 1;"DELETING": FOR
 i=count TO n-1: LET c(i,1)=c(i+
1,1): LET c(i,2)=c(i+1,2): LET c
(i,3)=c(i+1,3): NEXT i: LET c(n,
1)=0: LET c(n,2)=0: LET c(n,3)=0
: LET n=n-1: LET c(max,3)=n: GO
TO 6510
6640 IF i$="A" THEN  LET auto=NO
T auto: PRINT  OVER 1; INVERSE 1
;AT 21,0;"AUTO (A)"
6690 GO TO 6545
6700 PRINT AT pos,0,,AT pos,0; O
VER 1;TAB 6-LEN STR$ INT c(count
,1);c(count,1);TAB 16-LEN STR$ I
NT c(count,2);c(count,2);TAB 27;
c(count,3);TAB 29;CHR$ (150+c(co
unt,3))
6710 IF count>n THEN  LET n=coun
t: LET c(max,3)=n
6720 IF auto THEN  LET pos=pos+(
1 AND pos<20 AND count<max-1)
6730 RETURN
6997 REM **************
6998 REM **SAVE ARRAY**
6999 REM **************
7000 LET a$="Name of file to sav
e?": GO SUB 10: IF LEN z$>10 THE
N  GO TO 7000
7005 IF z$="" THEN  RETURN
7010 SAVE *"m";1;z$ DATA c(): RE
TURN
7497 REM **************
7498 REM **LOAD ARRAY**
7499 REM **************
7500 LET a$="Name of file to loa
d?": GO SUB 10: IF LEN z$>10 THE
N  GO TO 7500
7505 IF z$="" THEN  RETURN
7510 LOAD *"m";1;z$ DATA c(): RE
TURN
7997 REM **************
7998 REM **INITIALISE**
7999 REM **************
8000 BORDER 0: PAPER 0: INK 6: C
LS : LET max=300: DIM c(max,3):
LET e$=CHR$ 27: LET xdots=1200:
LET ydots=1080: LET xint=20: LET
 yint=20: LET x$="X axis": LET y
$="Y axis": LET t$="HIGH RESOLUT
ION POINT PLOTTING"
8010 LET xmax=100: LET ymax=100:
 LET xmin=0: LET ymin=0
8020 RESTORE 9500: FOR i=65368 T
O 65390: READ m: POKE i,m: NEXT
i
8025 RESTORE 9600: FOR i=65416 T
O 65447: READ m: POKE i,m: NEXT
i
8030 GO TO 1000
8997 REM ********************
8998 REM **DATA FOR MARKERS**
8999 REM ********************
9000 DATA 0,0,0,0,16,16,16,16,16
,16,124,124,16,16,16,16,16,16
9001 DATA 0,0,0,0,124,68,68,68,6
8,68,84,84,68,68,68,68,68,124
9002 DATA 0,0,0,0,0,0,0,56,56,56
,56,56,56,56,56,0,0,0
9003 DATA 0,0,0,0,16,40,40,68,68
,68,84,84,68,68,68,40,40,16
9497 REM ********************
9498 REM **MACHINE CODE DATA**
9499 REM ********************
9500 DATA 62,3,205,1,22,237,75,1
09,255,120,177,200,58,108,255
9505 DATA 215,11,195,97,255,0,0,
0
9597 REM ************
9598 REM **UDG DATA**
9599 REM ************
9600 DATA 0,0,16,16,124,16,16,0
9610 DATA 0,0,124,68,84,68,124,0
9620 DATA 0,0,0,56,56,56,0,0
9630 DATA 0,0,56,68,84,68,56,0
```

hensive facilities for editing data, and for modifying the final printed display to suit a wide range of applications. This complete SUPA-PLOT program is given in Listing 2. The only modification you may have to make is to line 6005. This is the "TOKENS OFF" command for the Kempston "E" interface, and you'll need to change it if you're using a different interface of course.

Do please be *very* careful when you type in the listing, paying special attention to lines 6000-6400. In particular, every one of the semi-colons embedded among the printer control codes needs checking here (and remember that I speak from experience — I have vast quantities of wasted paper lying at my feet to prove it!) Once you've typed it in, save it to auto-run from line 8000, and all is done. There's no need to load in the few bytes of machine code separately — there are so few that I've arranged for them to be poked in by the initialisation routines within the program itself.



Figure 1. Main menu.

So, you've loaded SUPA-PLOT in, and you're greeted with the main menu (Figure 1). What next? Let's take the options in the order in which you'll use them when starting from scratch:

1. Define axes.

Selecting this will present you with a screen like Figure 2, and you can now change the plotting parameters to suit whatever application you have in mind. Pressing "Y" will allow you to change the maximum and minimum values for the Y-axis (just follow the prompts) and also the interval for the scale divisions. (You might want division markers every five units, for example, or every 25 units — but for best results, whatever interval you use, check that it divides exactly into the complete range). At any time during this sequence, "ENTER"



Figure 2. Defining axes.

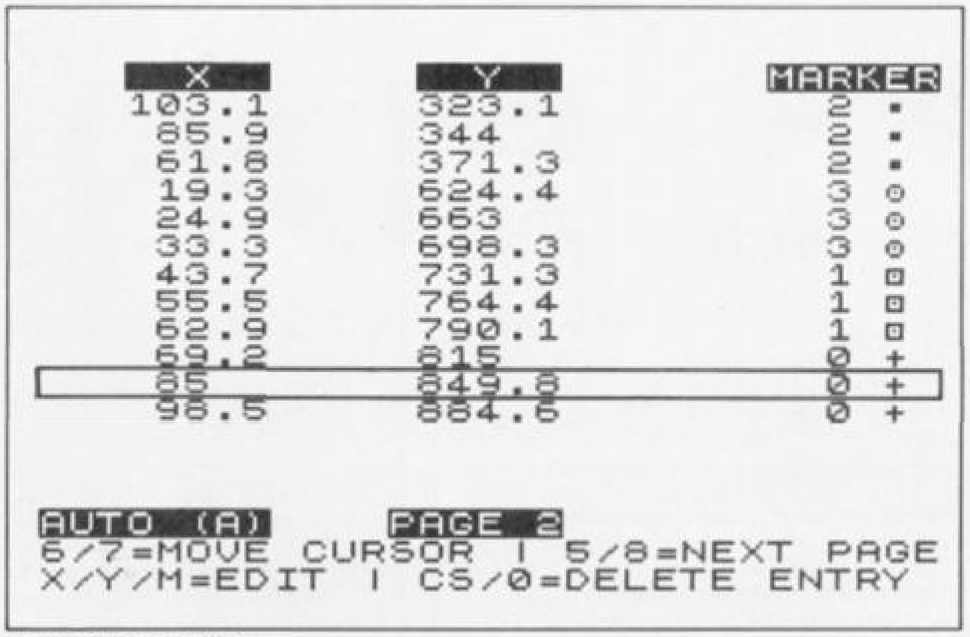


Figure 3. Data editor.

will cause the old value to be retained. Pressing "X" will allow you to make similar changes to the X-axis. If you press "T", the program will invite you to make changes to the text which will appear on your final printout. There are three such bits of text: labels for the X and Y axes (up to 16 characters only), and an overall title (unlimited in length, but I recommend that you don't exceed 64 characters). Finally, selecting "F" (for "format") will allow you to change the actual length of the axes in inches, if the default 5"x5" values aren't suitable for some reason. The Y-axis can be any length up to 8", and the X-axis up to 5" (whole numbers only).

Please note that only positive x and y values are handled by the program, and also that the program will not stop you from entering really silly values (such as making the maximum value for the X axis less than the minimum!). I've tested the program extensively without encountering any problems — but if you do accidentally crash it (at any stage — not just this one), GOTO 1000 will get you back into business without destroying any work you've already done. Once you've established suitable axis parameters, press "ENTER" to return to the main menu for the next step.

4. Edit data.

This option will offer you a screen like Figure 3 (but initially without the numbers of course!). To enter an "X" co-ordinate, press "X", and type in the number. In "AUTO" mode, the long red cursor will then automatically move down the screen to anticipate the next input. But you have a choice here: if you want to type in all the "X" values first, and then the "Y" values, just leave the program in AUTO mode (to which it defaults). But if you prefer to type the data in (x,y) pairs, one at a time, you can switch off the AUTO facility by pressing "A" (which toggles AUTO mode on and off). The sequence is then to press "X", enter the number; press "Y", enter the number; and then move the cursor down to the next line yourself.

At any time you can move the cursor up or down by using keys 6 and 7. So if you want to change an entry, just move the cursor to the appropriate position, press "X" or "Y", and the new value will be inserted in place of the old one. Pressing CAPS SHIFT/0 will delete the entire entry at the current cursor position and close up the gap — so use it with care!). Initially, all entries are made into "PAGE 1", but this will be full after 20 lines of data. No problem. Once Page 1 is full, you can turn to Page 2 by pressing key 8, and carry on entering data. The program will keep on generating new pages as they're needed, and you can switch between adjacent pages using keys 5 and 8 whenever you like, editing or deleting data wherever you wish. The order in which you type in the data points is irrelevant.

What about that last column, headed "MARKER"? Normally this will contain zeros, with a little "+" sign at the side. This indicates the kind of marker that will be used to plot the point on the final copy — in this case a simple cross. But there may be occasions when you want to plot several sets of data on the same graph, using different markers. Four are available, and by pressing "M" you'll be offered the option of changing the marker for the (x,y) pair at the current cursor position. The markers are numbered 0 to 3 thus: (0)=cross, (1)=dot within square, (2)=bold single dot, (3)=dot within circle. Just enter the number for the marker you want.

When you've entered your data, and chosen your markers, press ENTER to return to the main menu, and select:

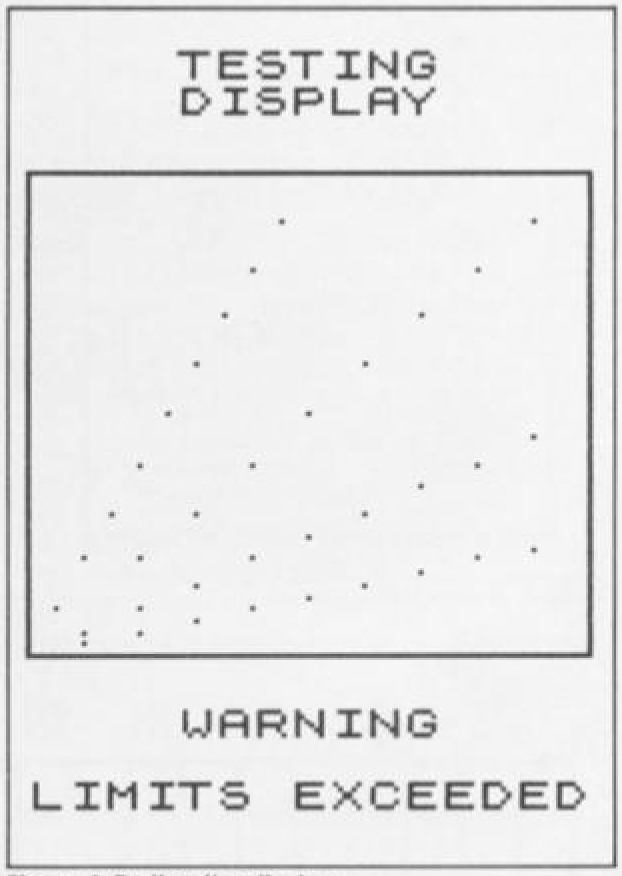


Figure 4. Testing the display.

2. Test display.

This will give you a screen like Figure 4, in which all your data points will be plotted within a rectangle. Think of this as a crude sketch of your final diagram, which gives you the chance to change your mind about the scales of the axes. If any of your points are outside the limits set by your maximum and minimum X and Y values, these points will not be plotted; instead a warning will be flashed on the screen. If this happens, or if you decide to change the maximum and minimmum values for some other reason, just return to the main menu and select option 1 again to make the necessary adjustments.

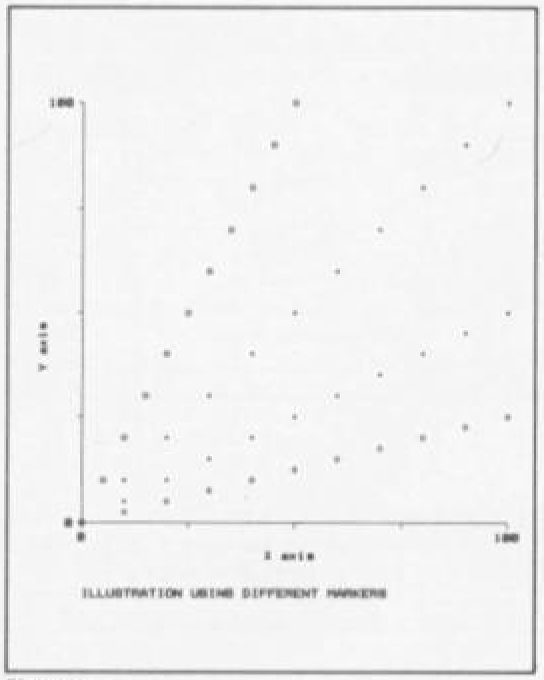


Figure 5.

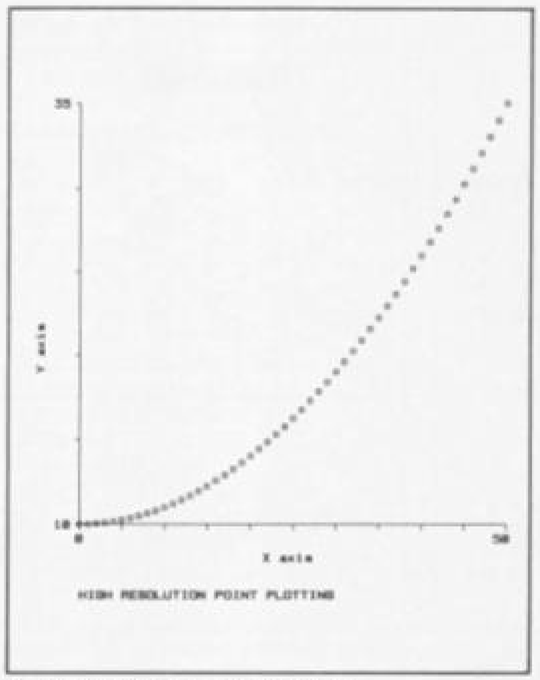


Figure 6. Plotting a parabola.

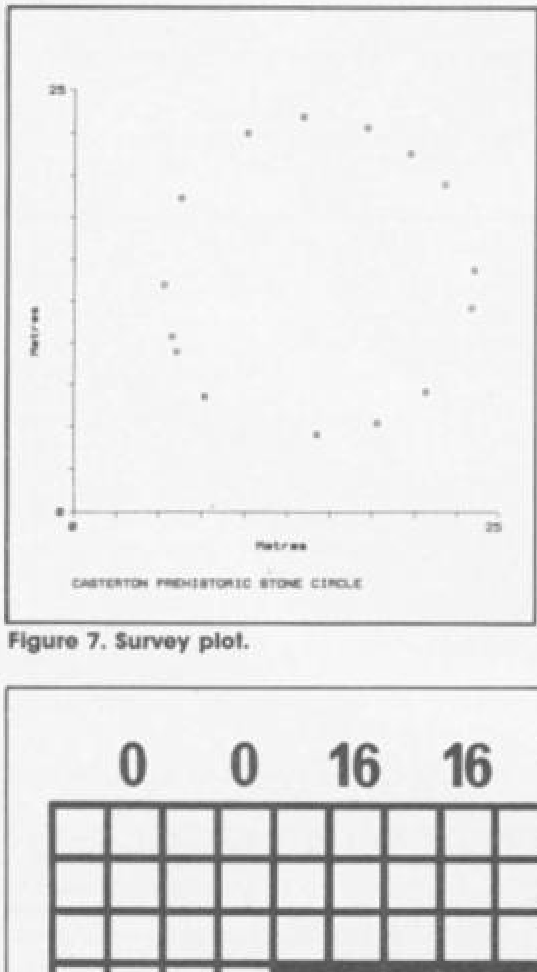


Figure 7. Survey plot.

When you're completely happy that your data are correct, that the markers are suitably chosen, and that the "TEST DISPLAY" option is performed without a flashing warning, you're all set to go. Return to the main menu and (trying NOT to hold your breath!) select option 3: "HARD COPY". The printer (you did remember to switch it on, didn't you?) should then become very busy for a couple of minutes. It will draw out your axes, mark divisions at the chosen intervals, plot the points, neatly label the axes in sensible positions, and print the title. It won't make you a cup of coffee while you wait — I'm still working on that.

To give you some idea of the possibilities, and an indication of the high quality of the final copy, Figure 5 shows the way in which the markers can be used to distinguish different sets of data points — in this case, the points lie on straight lines with different slopes. Figure 6 gives a set of points lying on a parabola, using different scales on the X and Y axes. You might also be interested to know that in both figures, the point at the intersection of the axes was the LAST point plotted. In other words, after more than 100 paper feeds back and forth, of varying size, the printer was still plotting with a truly astonishing accuracy, hitting the origin bang on the nose. As a final demonstration of the versatility of the program, and for those who are intrigued by outlandish uses of the Spectrum, Figure 7 is an exact plan of a prehistoric stone circle. As a severe test of precision, the whole ring was actually plotted out twice over in this diagram — but as you can see the second plottings coincided exactly with the first, with no discernible error.

Marker

One modification you may wish to make is to use different marker shapes. This is easily done by changing one or more of the DATA statements in lines 9000-9003 rather as you would for UDGs, except that instead of an 8x8 grid, it's 18x8 — and the numbers define the shape from left to right rather than from top to bottom (see Figure 8). The program assumes the "centre" of a marker shape to be the eleventh grid cell from the left, and the fourth from the top — so for a single tiny dot you'd need to make all eighteen numbers zero except the eleventh, which would be 16 (binary 00010000). Notice that a shape should be defined so that its width is about three times greater than its height if it's to appear in the right proprotions when printed.

Just a couple of extra points. Your data are stored in the array (c), which also contains the marker information. This array can be saved or loaded using options 5 and 6 on the main menu. If you think you're likely to want to plot more than 299 points, just alter the value of "max" to 500, or whatever, in line 8000.

Finally, what about continuous lines? At present, SUPA-PLOT will only plot points; it won't, for example, draw a "best fitting" line through them. If it could operate quickly enough, a facility for doing this would obviously be worth having — and when I get round to doing it, I'll let you know. Even without it, however, I'm already wondering whether I'll ever have a use for graph paper again . . .

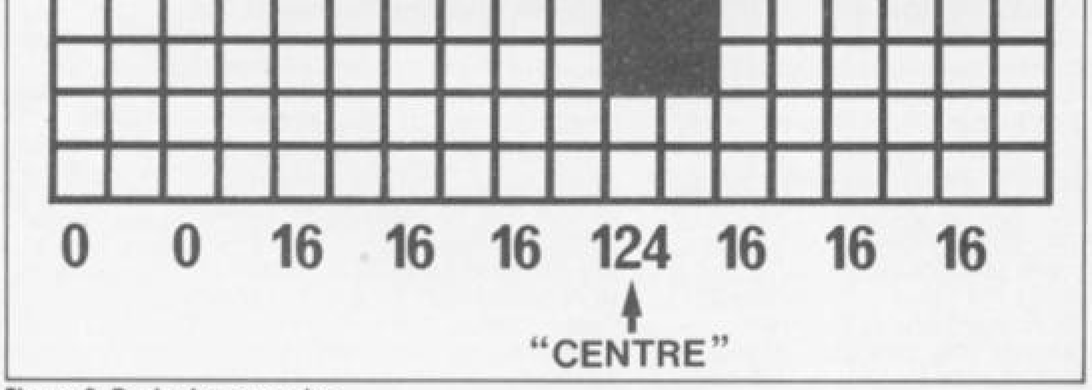


Figure 8. Designing a marker.

SUPA-PLOT

Specialist REPAIR Guide

Contact Nicola Baty on 01-437 0699

Ring Nicola Baty
on 01-437 0699
for details of
series discounts

NOBLES COMPUTER REPAIRS

- ★ All repairs carry a 4 month guarantee.
- ★ Free estimates.
- ★ 7 days a week, 24 hour answering service.

NOBLES
14-17 Eastern Esplanade,
Southend-on-Sea, Essex.
Tel: (0702) 63377/8

COMPUTER REPAIRS

Spectrum, Commodore, Amstrad,
Acorn, BBC etc.
Fixed charge repairs on all makes
Please ring for details
MCE SERVICES
33 Albert Street, Mansfield,
Notts NG18 1EA
TEL: 0623 653512

COMPUTER REPAIRS

We are the experts, having serviced Sinclair computers since the introduction of the ZX80.
Don't waste money on estimates — we repair Sinclair computers at price quoted (inclusive parts, labour, postage, VAT, irrespective of fault. No hidden charges).
Repairs guaranteed for 3 months.

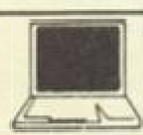
Spectrum	£18.75 inc parts
ZX81	£11.50 inc parts
16 KRam	£9.95 inc parts
Microdrive	£15.95 inc parts
Interface 1-11	£18.75 inc parts
also	
BBC	£22.00 + parts
Electron	£19.95 + parts
XK Memory Expansion Kit	£15.95

Computer Retailers please phone for Special Trade Price.
Call or send with cheque or P.O.
T.V. SERVICES OF CAMBRIDGE LTD.
French's Road, Cambridge, CB4 3NP.
Tel: 0223 311371

COMPUTER REPAIRS

Fixed Super Low Prices!
Inclusive of parts, labour and VAT

FIRST AID FOR TECHNOLOGY



1 week turnround Free collection/delivery available for local area

★ INTRODUCTORY OFFERS ★

SPECTRUM	£14 inc. + Free Game	ELECTRON	£19 inc.
C64	£22 inc. + Free Game	AMSTRAD 464	£32 inc. + Free Game
C16	£18 inc.		
VIC20, C+4	£22 inc.	C & N CASSETTE UNITS	£19 inc.
BBC	£32 inc.	C64 P.S.U. for sale	£20 inc.

Please enclose payment with item – 3 month warranty on repair.
Please enclose advert with repair

Tel: 0582 458375. **W.T.S. ELECTRONICS** Telex: 265871
5-9 Portland Road, Luton, Beds LU4 8AT.

SPECTRUM REPAIRS

£14.95 inclusive of labour, parts and p&p.
Fast, reliable service by qualified engs., average repair **24 hrs. 3 month guarantee.**
For help or advice ring:
H. S. COMPUTER SERVICES,
Unit 2, The Orchard, Warton, Preston,
Lancashire PR4 1BE. Tel: (0772) 632686

QUANTUM ELECTRONIC SERVICES

SINCLAIR SERVICE CENTRE
We have the spares for repairs to:
SPECTRUM/+/+2, QL, MICRODRIVE, INTERFACE 1.

- Fast repair service
- Free estimates
- While you wait service available in most cases. Please phone first.

OR IF YOU WANT TO DO IT YOURSELF!

4164 Memory	£1.90
4116 Memory	£0.99
Z80 A CPU	£2.95

All prices include VAT. Please add £1.00 p&p.
QUANTUM
33 City Arcade, Coventry, CV1 3HX.
Phone: Coventry (0203) 24632

SPECTRUM REPAIRS

48k Spectrum Faults	£17.00
48k Spectrum Keyboard Faults	£12.00
4164 Memory	90p
4116 Memory	50p
Z80A CPU	£2.10

All prices include VAT & Postage
R.A. ELECTRONICS
133 London Road South, Lowestoft,
Suffolk NR33 0AX.
Tel: 0502 66289

NEXT DEADLINE 19TH MAY

WHY NOT ADVERTISE IN OUR NEWEST SECTION + PUBLICISE YOUR REPAIR SERVICE? 01-437 0699

MICROPOWER

NORTHWOOD HOUSE
NORTH STREET
LEEDS LS7 2AA
Authorised Servicing & Repair Centre
TEL: (0532) 458800
A MSTRAD
A CORN
A TARI

SPECTRUM REPAIRS

From **£3-£19** incl. all parts, labour and 3 month Guarantee.
Phone for special postal rates
MICRO FARE
296 Gloucester Road, Horfield,
Bristol
Tel: 0272 46575

INTO THE 128

Toni Baker begins a series exploring the secrets of the 128, and starts by proving that you can switch between 48 and 128K modes at will.

Welcome to the first of a new five part series on the 128K versions of the Spectrum (ie both the Spectrum 128 and the 128+2) from a machine code perspective. In this series I intend to cover all aspects of the new machines where they differ from the old 16K and 48K machines. I'll be going through all the special features and explaining how you can access them from machine code.

First of all, the most obvious feature of the 128 is its large memory. In fact it now has 32K of ROM and 128K of RAM, making 160K in all — quite a phenomenal amount really. This memory is organised in what are called *pages*, a page being simply a chunk of continuous memory. On the 128 there are two pages of ROM and eight pages of RAM, and each page contains exactly 16K of memory. Hence we have 2 * 16 = 32K of ROM, and 8 * 16 = 128K of RAM altogether.

Traditional machine codists will be aware that on the 48K Spectrum the memory was continuous and occupied all addresses from 0000 to FFFF inclusive (with 0000 to 3FFF being ROM and the rest RAM). It is clear, however, that if we want to address more memory than was alowed on the 48K then four hexadecimal digits is simply not enough. To address the vast banks of memory on the 128 we need not four but FIVE hexadecimal digits, so we can have addresses like 12345h or 7CDEF. To see how all this works let's first look at the ROM.

A ROM with a view

There are two pages of ROM, and each page contains 16K. The first page is numbered (not surprisingly) from 00000to 03FFF inclusive. This is known as ROM zero, and may be colloquially referred to as 'The NEW ROM'. When the machine is first switched on or the RESET button pressed then control begins at the very start, at address 00000. This 'New ROM' is in fact totally new and is not present on 16K and 48K machines.

The second page of ROM is numbered from 10000h to 13FFF. This is called ROM one, and is sometimes referred to as "the OLD ROM". Although a few changes have been made this is, essentially, the original ROM which was present on 16K and 48K machines. Notice that the numbering of the ROMs is not continuous (ie there is a gap between the end of ROM 0 (03FFF) and the start of ROM 1 (10000h)).

The next task is to see how all this ties in with machine code.

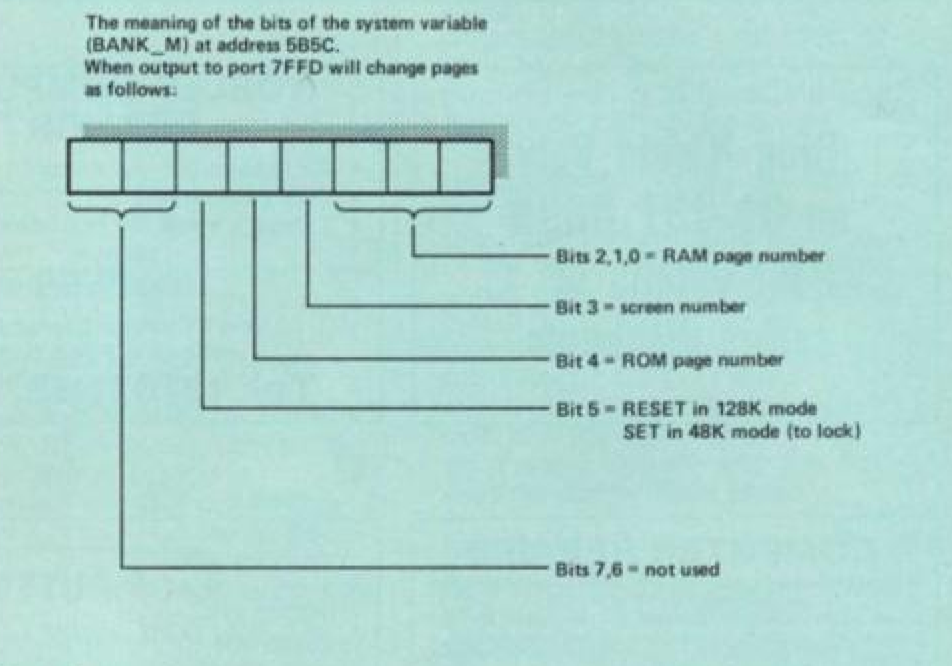


Figure 1.

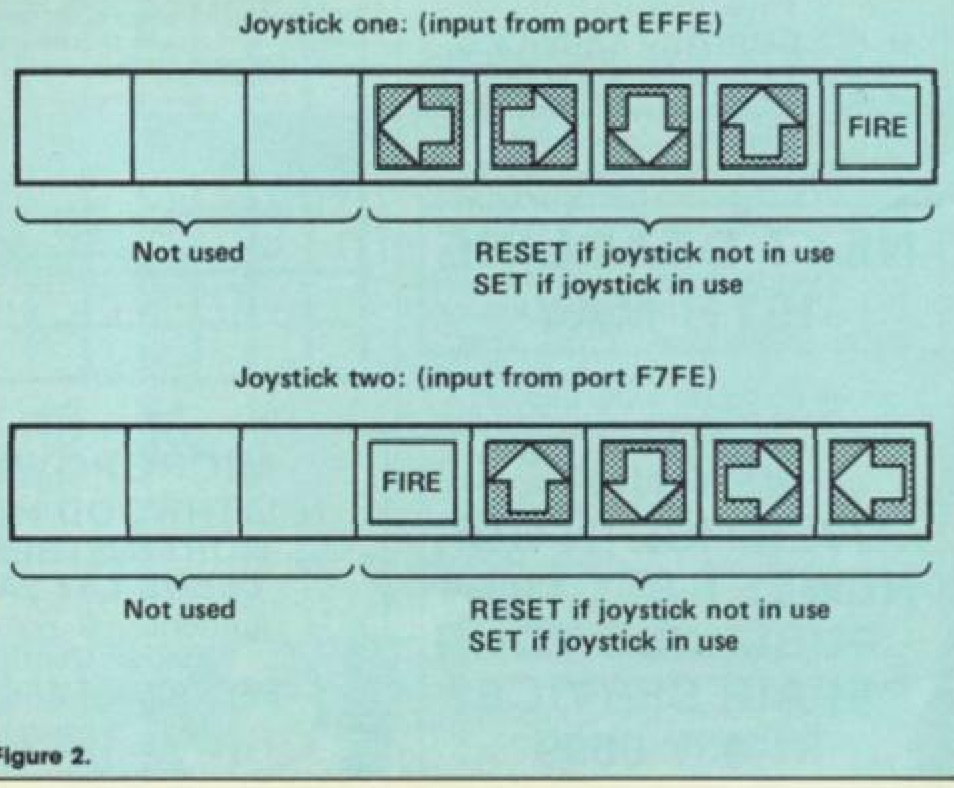


Figure 2.

You are probably aware that all machine code instructions are designed for an address range using only four hex digits, not five. For instance **CALL xxxx** is allowed but **CALL xxxxx** is not. Instructions like **LD A,(HL)** or **JP (HL)** may at first glance seem like a way round the problem, but then you have to remember that HL itself (as with all register pairs) can only hold values between 0000 and FFFF, which leaves us back at square one.

The solution is actually very simple. An instruction like **CALL 1234h** can mean *either* CALL 01234h" *or* CALL 11234h" depending on which of the ROMs is *paged in*. You can visualise this by imagining that only one of the two ROMs is allowed in at a time. Either of the ROMs (but never both) may be present in memory at any one time. If ROM 0 is present then we say that ROM 0 is *paged in*. If ROM 1 is present then we say that ROM 1 is paged in. Whichever of the ROMs is paged in (since one of them must be at all times) is referred to as the *current* ROM, or the current ROM page. Thus the current ROM will invariably be either 0 or 1 — this supplies the first digit all ROM addresses. The remaining four digits may be handled in machine code as normal.

Since there are only two ROMs and no more, addresses such as A1234 cannot exist. I have, however, heard of the Shadow ROM of the ZX Interface One being referred to as ROM two, in which case maybe addresses 20000h to 21FFF could refer to the Shadow ROM, but this is non-standard so I won't go into it in any detail.

On the RAM PAGE

Having dealt with ROM addresses it is now time to deal with RAM. Addresses in RAM are all between C000 and FFFF, and as with the ROMs, only one page of RAM may be paged in at any one time. The pages of RAM are numbered from zero to seven, so this means that RAM addresses all lie in one of the ranges 0C000 to 0FFFF, 1C000 to 1FFFF, 2C000 to 2FFFF, 3C000 to 3FFFF, 4C000 to 4FFFF, 5C000 to 5FFFF, 6C000 to 6FFFF, or 7C000 to 7FFFF. Whichever of the RAM pages is paged in is referred to as the *current* RAM, or the current RAM page. The current RAM page will be a digit between 0 and 7 — this supplies the first digit of all RAM addresses. The remaining four digits may be handled in machine code as normal.

Having established that all addresses between x0000 and x3FFF are in ROM, and that all addresses xC000 to xFFFF are in RAM, (where x refers to the relevant page number) this leaves us with a gap. What about addresses 4000 to BFFF?

The answer is a kind of shorthand. The address range between 4000h and BFFF is split up into two 16K chunks, ie 4000h to 7FFF and 8000h to BFFF. The first chunk is shorthand for RAM page five, and the second chunk is shorthand for RAM page two.

In other words, 4000h means *precisely the same thing* as 5C000. (Also, 5000h means 5D000, 6000h means 5E000 and 7000h means 5F000, etc). This means that, in effect, page five is permanently paged in, though with a modified addresses range.

In similar fashion, 8000h means the same thing as 2F000, etc). So page two is also effectively permanently paged in, again with a modified address range.

As well as being a shorthand method of accessing two of the

```
                            ORG 8000
F5          SWAP_2          PUSH AF
C5                          PUSH BC                 Stack A,B,C and flags.
01FD7F                      LD BC,7FFD              BC:= port number required for paging.
3A5C5B                      LD A,(BANK_M).          A:= current pages etc.
E61F                        AND 1F                  Reset bit five, to signal "unlocked".
C3085B                      JP 5B08,SWAP+8          Jump to complete SWAP routine.

                            ORG 800D
ED7B3D5C    MODE_48         LD SP,(ERR_SP)          Empty machine stack down to
                                                    error return address.
E1                          POP HL                  Drop error return address.
210313                      LD HL,1303,MAIN_4
E5                          PUSH HL                 Stack 48K mode error return address.
21761B                      LD HL,1B76,STMT_RET
E5                          PUSH HL                 Stack return address from
                                                    statement execution.
2A455C                      LD HL,(PPC)             HL:= current line number.
22425C                      LD (NEWPPC),HL          Store as next line number.
3A475C                      LD A,(SUBPPC)           A:= current statement number.
3C                          INC A
32445C                      LD (NSPPC),A            Store statement number of next
                                                    statement to interpret.
0600                        LD B,00                 Bit 4 reset signals "48K mode".
11BE15                      LD DE,15BE              DE: point to channel "P" info (48K).
1810                        JR MODE_CHANGE          Jump to change mode.

                            ORG 802E
3EC3        MODE_128        LD A,C3
32005B                      LD (SWAP),A             Store a JP instruction at SWAP.
210080                      LD HL,SWAP_2
22015B                      LD (SWAP+1),HL          Set JP destination as SWAP_2.
0610                        LD B,10                 Bit 4 set signals "128K mode".
115480                      LD DE,8054,P_128        DE: pts to channel "P" info (128K).
3A3B5C      MODE_CHANGE     LD A,(FLAGS)
E6EF                        AND EF                  Reset bit 4.
B0                          OR B                    Assign bit 4 according to mode reqd.
323B5C                      LD (FLAGS),A            Store revised flags.



2A4F5C                      LD HL,(CHANS)           HL: points to base of channel
                                                    information area.
010F00                      LD BC,000F
09                          ADD HL,BC               HL: points to info for channel "P".
EB                          EX DE,HL                DE: points to info for channel "P".
0E04                        LD C,04                 BC:= number of bytes to copy (=4).
EDB0                        LDIR                    Copy data into channel "P" info.
C9                          RET                     Return.

                            ORG 8054
345B        P_128           DEFW 5B34,POUT          Channel "P" output address.
2F5B                        DEFW 5B2F,PIN           Channel "P" input address.
```

Assembler listing for program to switch between 48 and 128K modes.

eight RAM pages, we also now have an illusion of continuity — and this is the marvellous trick which makes the 128 compatible with the 48K machine. You see, we now have the situation whereby every single (four digit) address from 0000 all the way up to FFFF now actually means something. At every such address there is memory available, with 0000 to 3FFF being ROM and the rest being RAM. We now have the appearance of a continuous stretch of RAM from 4000h all the way up to FFFF. This is a 48K illusion, but it makes the thing compatible with other Spectrums, and is jolly useful for machine codes.

Page to Page

The next thing you'll need to know is how to change pages — that is — how to specify which ROM page and which RAM page is to be paged in. The answer comes in the form of an OUT instruction. Output port number 7FFD is the key to the thing. You can think of it almost like an extra register or memory location, in that it can store a value between 00 and FF. Like a memory location you can assign it with a new value, but, *unlike* a memory location it is impossible ever to read its value.

There is essentially only one way of writing to this port in machine code. The sequence of instructions:

```
LD BC, 7FFD
LD A,??
OUT (C),A
```

(You can of course swap the order of the first two instructions).

It is this port which PAGES IN the various ROM and RAM pages. Take a look at figure One. This shows the meanings of the individual bits of port 7FFD. As you can see, bits two to zero specify the RAM page while bit four specifies the ROM page. I'll come to the meanings of the other bits later on.

There is, of course, one major problem with port 7FFD. Because you cannot read from this port it is impossible to actually determine which ROM or RAM page is currently paged in at any moment in time. For this reason the Spectrum 128 keeps a back-up copy of the value of port 7FFD in one of its new system variables. The system variable is called BANK-M. Its address is 5B5C. Because of this back-up copy reading (BANK-M) will tell you what is paged in and what isn't. It also means, however, that whenever you change page by writing to port 7FFD then you must also store the value in BANK-M, or else the ROM may cause a crash.

In fact, the Spectrum 128's interrupt routines (which lie at addresses 00038 and 10038) actually require that bit four (at least) of BANK-M be accurate. This means that interrupts *must* be disabled whilst changing ROMs. Furthermore, on leaving the interrupt routine the value in (BANK-M) is output to port 7FFD, so if they didn't match before, they certainly will afterwards. The upshot of all this is that if you want to change RAM page (or screen — but more of that later) then *either* interrupts must be disabled throughout the change, *or* (BANK-M) must be loaded first and port 7FFD loaded second. If you get this the wrong way round then you'll get a crash the first time an interrupt occurs between the two instructions.

Screen display

As with the 48K Spectrum the screen is stored at address 4000h. As I've already stated this is the same thing as 50000 (ie address C000 on RAM page five). The Spectrum 128, however, has not one but *two* screens, though this isn't obviously mentioned in the instruction manual.

Both of the 128's screens are memory mapped, but obviously only one of them can ever be visible on the TV at any one time. The normal screen which appears all of the time under normal circumstances is called SCREEN ZERO, but there is also an alternative screen, called SCREEN ONE, which is stored at address 7C000 (ie addresss C000 on RAM page seven).

If one of these screens is visible on the TV then it is said to be ACTIVE. Normally screen zero is active at all times and screen one is never active, however you can change this from machine code by setting bit three of (BANK-M) and output port 7FFD (see Figure One). What's more, it makes no difference if a different RAM page is paged in. For instance, screen one is stored on RAM page seven. Suppose we made screen one active by setting bit three of (BANK-M) and subsequently outputting the value to port 7FFD. This would meant that the contents of screen one would be visible on the TV. The pixels of screen one are stored at addresses 7C000 to 7D7FF, whilst the attributes are stored at addresses 7D800 to 7DAFF. It is clear that page seven must be paged in in order to change the contents of screen one, however it is *not* necessary for page seven to be page in in order for screen one to be active. *Any* RAM page may be paged in, and screen one would still be visible as long as it remained active. Only by activating screen zero can screen one be made to vanish from the TV.

Surprisingly, it turns out that you *can* change the current screen in BASIC, without having to use machine code at all. The BASIC instruction **POKE 23388,24** will activate screen one, whereas **POKE 23388,16** will activate screen zero (and so deactivate screen one). If you try this then all you'll get is a black TV picture since all the attribute bytes contain zero. Unfortunately it is not possible to print anything onto screen one from BASIC — machine code is a must here. Still — the technique can be useful. If your BASIC program contains the three instructions.

POKE 23388,24: LOAD "" SCREEN$: POKE 23388,16

Then the TV will appear to go blank whilst a file is loaded from tape. When the loading is complete the finished picture will appear on the TV instantaneously!

Another useful point to note is that any error report message generated by a BASIC program (including *0:OK*) will automatically re-activate screen zero, so if you forget to switch screen one off then the Speccy will do it for you.

Port

There is one more bit from port 7FFD which is used. This is bit five, which in 128K mode must always be RESET. If this bit is set then the machine 'locks' in 48K configuration (ie with ROM 1 permanently paged in, RAM page zero permanently paged in, and screen zero permanently active). Setting bit five of (BANK-M), however, is not in itself sufficient to enter 48K mode, because the machine stock will contain a few "return addresses" which were intended to refer to ROM 0, and once the machine is locked ROM 0 cannot be paged in.

The 128 contains two new BASIC commands — PLAY and SPECTRUM. The more complicated PLAY I shall leave until part five of this series, but SPECTRUM I can deal with now. The effect of typing the command SPECTRUM is twofold — firstly, the machine goes into 48K mode, and secondly the error report *0:OK* is produced, stopping any program in its tracks.

Page thirty of the Spectrum 128 +2 manual tells us, and I quote, "Once in 48K BASIC mode, there is no way back to 128 BASIC mode apart from resetting the +2 (or switching off, then on again)." The same is also true of the ordinary Spectrum 128. The

program which accompanies this article, however, will do just that! It is now possible to switch between 48K mode and 128K mode at will! Here's how it's done:

Mode switching

First, switch the machine on (or reset it) and select "128 BASIC" from the main menu. You may then type **SPECTRUM** if you wish, but the program will not work if you select "48 BASIC" from the main menu. **CLEAR 32767** will ensure that the program remains safely above RAMTOP. LOAD the program which I have listed. You now have two additional commands at your disposal:

Randomize USR 32781

is similar to the command **SPECTRUM,** in that it will cause the machine to enter 48K mode, however it will cause the machine to enter 48K mode, however it will do so *without* generating the report *O:OK,* so that program execution may continue from the next statement. To complement this the command

Randomize USR 32814

will re-enter 128K mode, and will again continue from the next BASIC statement. Whilst all this sounds nice and cozy, please bear in mind that there are rules to follow — NEVER use the command **COPY** whilst in 48k mode, NEVER use **LPRINT** or otherwise print to channel "P" (normally stream three) whilst in 48K mode, and NEVER place any code into the printer-buffer. The reason for these restrictions is that the printer buffer is used by the Spectrum 128 to store new system variables and subroutines. Corrupting the printer buffer would truly make a return to 128K mode impossible.

This new arrangement, however, has tremendous advantages for people who own a Spectrum 128 with a real printer plugged into the RS232 socket — especially if they don't own an Interface One - for it now becomes possible to use the printer in 48K mode after all.

Firstly you must set the BAUD rate. This can only be done in 128K mode using the command FORMAT "P";N. Then you can type either **SPECTRUM** or **RANDOMIZE USR 32781** to enter 48K mode. Now to print anything onto your printer all you have to do is include the following statements in your program:

RANDOMIZE USR 32814: LPRINT (anything): RANDOMIZE USR 32781

As it happens, the program is relocatable too, except that there is one absolute reference to the label SWAP-2, and another to label P-128. Both of these must be changed if you wish to relocate the program.

There is just one other aspect I want to discuss in this article. The Plus 2 comes complete with two little joystick sockets, and it is useful to know how to read the joysticks from machine code. To read from JOYSTICK 1 you must input a byte from port EEFE. Figure two will show you how to interpret that byte. Each bit will normally be set, but will be reset if the joystick is in use and corresponds to the correct direction.

To read the JOYSTICK 2 you must input a byte from port F7FE. Figure two will again show you how to interpret that byte. The machine code sequence LD BC,F7FE/IN A,(C) will input the byte into the A register.

In the next article in this series I shall tell you all about page codes, and the Spectrum's "Silicon disc" file storage system. See you then.

ZX COMPUTING

FULL CREDIT FACILITIES FOR ACCESS AND BARCLAYCARD.

Lineage: 48p per word. (VAT inclusive)

Semi display: £9.50 per single column centimetre + VAT. Ring for information on series bookings/discounts (minimum 2.5cm).

All advertisements in this section must be prepaid.
Advertisements are accepted subject to the terms and conditions printed on the advertisement rate card (available on request).

01 - 437 0699

Send your requirements to:
NICOLA BATY
ASP Ltd., 1 Golden Square,
London W1

UTILITIES

NEW TAPE UTILITY 9

This program gives a tape back-up of most software. Even converts most jerky loaders to normal for reliable loading. Manages very long programs (even 51k!), fast loaders, etc, etc. State type of Spectrum when ordering (48k, 128k etc). **£6.99 on tape (£4.99 if updating)**

NEW TRANSFER PACK 3 – FOR TAPE TO DRIVE

A software system to convert many TAPE based programs to your DRIVE (no matter what type) + OTHER UTILITIES. Most jerky programs are EASILY converted to drive. Pack has at least 7 programs! Opus/Beta owners add £1 for OPTIONAL extra program. **State TYPE of Spectrum + name of DRIVE when ordering** (eg 128k and Microdrive). **£11.99 on TAPE OR £12.99 on m/drive cartridge.** (£6 if updating).
INFORMATION SHEETS – £1 each – approx 8 transfers of popular programs per sheet – needs TP3. Up to No 25 available. **SPECIAL OFFER – No's 1-20 only £4.50.**
MICROTRANS – for m/drive to m/drive back-up + tape to drive (inc headerless) – does not convert programs. Includes MICROTAPE and RUN programs. ONLY £3 on TAPE. £4.50 on Cart. For any Spectrum.
CODE COMPRESSOR – £3.50
Overseas: add £1 Europe, £2 others each product. SAE for details.
NO RISK! – ALL PROGRAMS CARRY OUR MONEY BACK GUARANTEE (not updates)

LERM, DEPT ZX3, 11 BEACONSFIELD CLOSE, WHITLEY BAY NE25 9UW (Tel: 091-2533615)

BRADWAY SOFTWARE (ZX)

LETTA-HEAD PLUS The most versatile Spectrum utility program for designing and printing business and personal stationery, receipts, letterheads, labels, posters, etc. Create the design on screen using inbuilt graphics or load any SCREENS from an arts package, add text from the library of 25 different fonts, select the required output format and print all the copies you need. **Price only £9.00**

DUMPY 3 Fast becoming the de facto standard screendump program, producing all the screen dumps you will ever need for your Spectrum. No need to understand assembler; **DUMPY's code GENERATOR** creates the screendump, relocates it, and saves it ready for use in your programs. Over 1000 different size, shape and density screendumps, anywhere in RAM! Price less than 1p per dump!! **£8.50.**

LIN-O-TYPE Add style to your written word. Print out on any wordprocessor file on tape, microdrive or disc in high denstiy NLQ in a choice of 25 fonts, or turn your Spectrum into a full WYSIWYG electronic typewriter, superb for addressing envelopes, filling in forms or writing short notes. **Price £8.50.**

ASTRUM - NEW!! The very best Z80 assembler and monitor package for the Z80 assembler and monitor package for the Spectrum: Exceedingly versatile, powerful and easy to use; Full wordprocessor style editor; Unlimited source code; Assembly using microdrives or the Opus Discovery ; Command driven monitor with multiple breakpoints, single stepping, slow run etc. **Price £13.50** (state microdrive or Opus version).

DISC MANAGER NEWEST!! A brand new utility for users of the Discovery drives; Full disc sector editor and file mangaement program; Erase, rename, copy goups of files; Examine, modify and understand disc directory information; Examine, modify and recover damaged date. **Price £10.50** including 3½" disc only.

Bradway Software programs drive most (currently 16 different) interfaces: **Letta-Head, Dumpy & Lin-O-Type** require an Epson compatible printer and are compatible with Microdrive, Wafadrive & Disc. Prices are for cassettes, include full instruction and P&P within Europe; please add £2.00 per program worldwide airmail postage, £1.50 for microdrive cartridge or disc.
Send for further details!

33 Conalan Avenue, Sheffield S17 4PG

ACCESSORIES

MAKE SOME SPECTRUM CeNCe
Simple Computer Controller for Machinery and Models using your Sinclair Spectrum.
A little black box which plugs into the expansion socket of any Spectrum and enables you to drive two small to medium stepper motors (up to 2 AMPs per phase) or up to 8 DC motors, relays, lamps or any other DC device.
Simple to Program and built by an engineering firm for its own use. You can use it for CNC turning, milling or coordinate drilling.
Reliable Prototype still running after three years use.
We provide interface instructions, software, hints and examples. Price £125 inc. carriage and VAT Power supplies and Stepper Motors available. We will be pleased to quote for larger units.
P.R. PRODUCTS, PARK MILL, HOBSON STREET, MACCLESFIELD, CHESHIRE SK11 6BE.
Tel: (0625) 22800.

RGB OUTPUT?
COMPOSITE VIDEO?
GREEN SCREEN?
TTL - ANALOGUE?
COMP SYNC?
BRIGHT?
adapt electronics

CONFUSING ISN'T IT?
Talk to the problem solvers. We have the experience and expertise to advise on all aspects of computer display problems, eg:
- **RGB OUTPUT MODULE** for Spectrum 16 or 48K. Much acclaimed - gives monitor quality pictures - no dot crawl £36.95
- **BRIGHT OUTPUT MODULE - RGBI TO RGB LIN.** - Allows full use of the bright facility on analogue input monitors and monitor TV's to produce full colour palette. Only for Spectrum 128 .... £29.95
- **MONITOR CABLES** - Connecting cable - any monitor/computer combination £5.00

Add P&P £1 (UK). £2 (EEC). £4 (others) on any order.
Send SAE for our Free Monitor Fact Sheet

20 STARLING CLOSE, BUCKHURST HILL, ESSEX IG9 5TN. Tel 01-504 2840.

RING 01-437 0699 FOR DETAILS OF OUR SERIES DISCOUNTS

WANTED

WANTED: YOUR non working Commodore Computers and equipment for cash. Ring D.M.R. Electronics, Bradford, (0274) 882892.

PRINTER SERVICES

FABRIC RIBBON RE-INKING
Introductory Offer
£1.50
Send ribbons and prepayment to:
ALADDINK
(Dept ZXC), 4 Harker Crescent, Eyemouth, Berwickshire TD14 5AP Tel: 08907 50965
Colour re-inking now available
See 'NEWS' ZX April 1987

SERVICES

STOP PLAYING GAMES
Use your computer to make money. Turn your hobby into a home-based income. Full and part time opportunities to cash in on this tremendous market. High earnings easily possible. Open to any amateur micro user and gamer. Write for free details.
Westlink Promotions (ZX)
108 George Street
Edinburgh EH2 4LH.

DEADLINES

19TH MAY
JULY ISSUE
15TH JUNE
AUGUST ISSUE
20TH JULY
SEPTEMBER ISSUE

WARNING NOTICE

Advertisements placed in this magazine are to be in strict compliance with our standard conditions (copies of which conditions are available on request) and on the clear understanding that the advertiser warrants that his advertisement(s) does not infringe any copyright or condition of sale of any interested party in the advertised product.

Further, the advertiser indemnifies the proprietors of this magazine in respect of costs, damages, or any other claims brought against them as a result of legal action arising from the publication of the advertisement.

Any breach of these terms or the said conditions may result in prosecution of the advertiser by the proprietors.

SOFTWARE

USE THIS SPACE FOR YOUR SUMMER PLANS 01-437 0699

TERMS & CONDITIONS

CLASSIFIED ADVERTISING TERMS & CONDITIONS

Our terms for new advertisers (semi-display and lineage) are strictly pro-forma payments until satisfactory reference can be taken up (excluding recognised advertising agencies). Cheques and P.O.'s should be crossed and made payable to ARGUS SPECIALIST PUBLICATIONS LTD. and sent together with the advertisement to:

"The Classified Dept.,
No. 1 Golden Square,
London W1R 3AB.

There are no reimbursements for cancellations. Advertisements arriving too late for a particular issue will be inserted in the following issue unless accompanied by instructions to the contrary. It is the responsibility of the advertiser to ensure that the first insertion of every series is published correctly, and corrections must be notified in time for the second insertion, otherwise the publishers will not accept any liability or offer any reduction in charges.

All advertising sales are subject to Government Regulations concerning VAT. Advertisers are responsible for complying with the various legal requirements in force eg: The Trade Description Act, sex discrimination act & the business advertisements (disclosure) order 1977.

Full Terms & Conditions of Advertising available on request.

AN OFFER TO SINCLAIR USERS FROM ALLAN DATA

We converted **simple** mathematical procedures and probability routines into a **best-selling** football forecaster — **"Form-Finder"** for users of the BBC range.

We are offering to a **limited** number of Sinclair users the opportunity to purchase the **same Forecasting Data** which you can use to **write your own** football result Forecaster for **your Sinclair**.

You can **sell as many copies of your completed program as you like, no royalties** or any further monies will be claimed by us after initial purchase.

NOTE: You would need only to sell 2-3 copies to recover the initial purchase price.

This **unique** + **genuine opportunity** to make cash will only be offered to Sinclair users during 3 specified fortnightly periods **per year**.

For full details **send a first class S.A.E.** to:

A. O'Brien, 1 Lorn Road, Dunbeg, Connel, Argyll PA37 1QG.

We request that persons under 18 yrs do not apply.

Tel: 0631 62923 10am — 2pm, Tues — Thurs

CHEZRON SOFTWARE
(Dept. 303) 605 Loughborough Road
Birstall LEICESTER LE4 4NJ

BONABANNER — Programmers! Unleash the power of string graphics on your Spectrum. Create title pages, text adventure graphics, program headers etc. that print instantly from Basic or machine code.

At just £3.50 why hesitate! — if our demo program doesn't convince that **BONABANNER** is useful to you, JUST RETURN THE WHOLE PACKAGE FOR A FULL REFUND!!

FILL THIS SPACE FOR AS LITTLE AS £24.45 PER MONTH + VAT

HUMPHREY SOFTWARE
TWO PROGRAMS

SUPER HANDICAP; The connoisseur racing program that will pay for itself. Full instructions and recommendations.
ONLY £9.95.

SEEK & FIND; Try to sink your opponents ships before he gets yours, but watch out for the mines! One or two players, many features.
ONLY £3.99

SPECIAL OFFER
BOTH PROGRAMS AT ONLY £11.95 INCL.

HUMPHREY SOFTWARE
8 Richard Andersen Court,
Monson Road, London SE14 5EJ.

STUDYING FOR THEORY of music exams? Try my practice programs! (Grades 2-5). Cassette £5, Microdrive (2 cartridges) £8. Ted Kirk, 33 Humber Crescent, St. Helens, Merseyside WA9 4HD. Details free.

QL/SPECTRUM UTILITIES! Wide range of business/practical programs available. SAE/IRC for details (state Micro). S.D. Microsystems (ZX) PO Box 24, Hitchin, Herts.

ARE YOU A BUDDING PROGRAMMER?

ZX is always looking for top quality games and utilities for publication. If you have a top notch game or a useful utility for the Spectrum or QL why not send it to us for appraisal on cassette or microdrive complete with a listing if possible.

There is also our new feature Short Cuts to showcase your practical, novel or imaginative short routines with cash prizes for published listings. For longer programs we pay competitive rates, and if you have an idea for an article or series for ZX — drop us a line or phone Bryan or Cliff on 01-437 0626 to talk it over.



which disc?

Thinking of changing to a disc system? John Wase weighs up the options.

Recently, ZX has had a lot of information about discs. We've not gone disc mad, but clearly there's a lot of interest here. You, the reader. however, might not possess a disc system. Just a microdrive, perhaps? Or merely a battered tape recorder? So, how do they work, and what's the advantage anyway?

A disc is a mechanical device, rather like a cross between a record player and a tape recorder. Thus it has a rapidly rotating table, and a reading head (sometimes two, one on each side), rather like a record player, and the discs themselves, although looking more or less like records, are coated with a brown magnetic material like a cassette tape. And there the analogy ends. For a record player records its information *sequentially* on one long spiral groove, just as a tape recorder records on one piece of tape, from one end to the other. A disc unit, on the other hand, marks out a series of concentric circles on the disc, dividing each circle up into smaller segments by putting down magnetic "marks". Its reading head can then jump to any part of these circles or tracks and read what's there. Instantly. Quite unlike a tape recorder, where one has to spool through the whole lot if one is looking for a particular program.

Bit-mapped

How does it do this? Well, it keeps a list, rather like a telephone directory, usually on the first track. This tells it where to look. I can best explain this further by taking two well-known examples, the Opus Discovery and the Disciple. In the latter, the Directory, or catalogue, is bit-mapped. What on earth does this mean? Well, it means that there is a mark for each of the sectors on each of the tracks, and that when you save a program, the name is added to the directory, along with markers indicating the sectors in which the program has been saved. After the disc has been in use for some time, and a lot of programs have been saved or erased, the free sectors available for the next save could well be scattered at random all over the disc!

Discovery's directory, on the other hand is not bit - mapped. This means that each program is saved on sequential sectors, a much simpler system. Moreover, it has the advantage that if you accidentally erase a program, (erase merely wipes the name off the catalogue file), you can recover it without too many problems, for it will all be saved sequentially, on adjacent sectors of the disc. Furthermore, it's pretty easy to increase the 110 files in the standard catalogue to cope with special needs. However, it does mean that with use, the disc gets useless small spaces left all over it; (the program is always saved in the biggest available free space), and eventually the disc has to be compacted (the programs all moved to the beginning) to release the space and allow you to save files on it. Conversely, you never need to compact Disciple discs, because Disciple uses every sector, but extending Disciple's catalogue is difficult. Horses for courses.

How does all this nonsense help? Well, the directory is like a tape counter but much more sophisticated. So you get *instant* access. You can load a program, a game for instance, with a main section and a lot of data tables, and change these during the course of the game. Instantly, without having to spend time searching for anything. You can, on a more sophisticated basis, rapidly access a file full of information and pick out *one item*, through use of random access files — just one name and address, maybe, and display it. From a disc containing thousands. Instantly!

This isn't the only advantage. You have to turn a tape recorder on and off, but a disc unit is controlled by software, like a microdrive. So it will load in bits as required by the software. Discs are fast. The Discovery is relatively slow, a little quicker than a microdrive, the Disciple about five times as quick. So for a 48K full load, sample times would be of the order of 20-25 seconds and 5 seconds respectively. And if you use a tape, you'll have to agree that that's very fast. But, overriding all is the *reliability* of discs: they are *so* reliable it's unbelievable, and if you haven't used them before, it's a revelation.

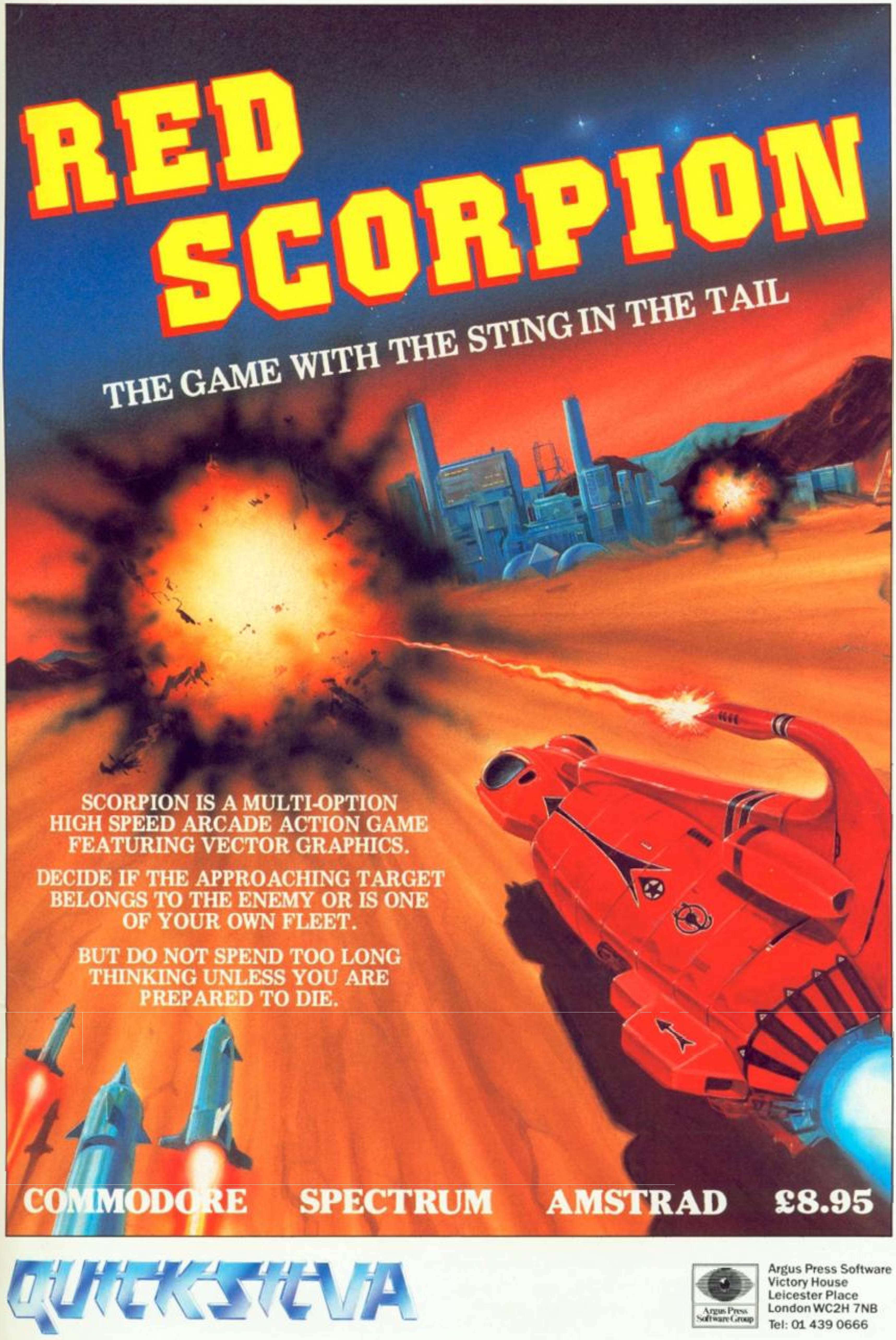
Options

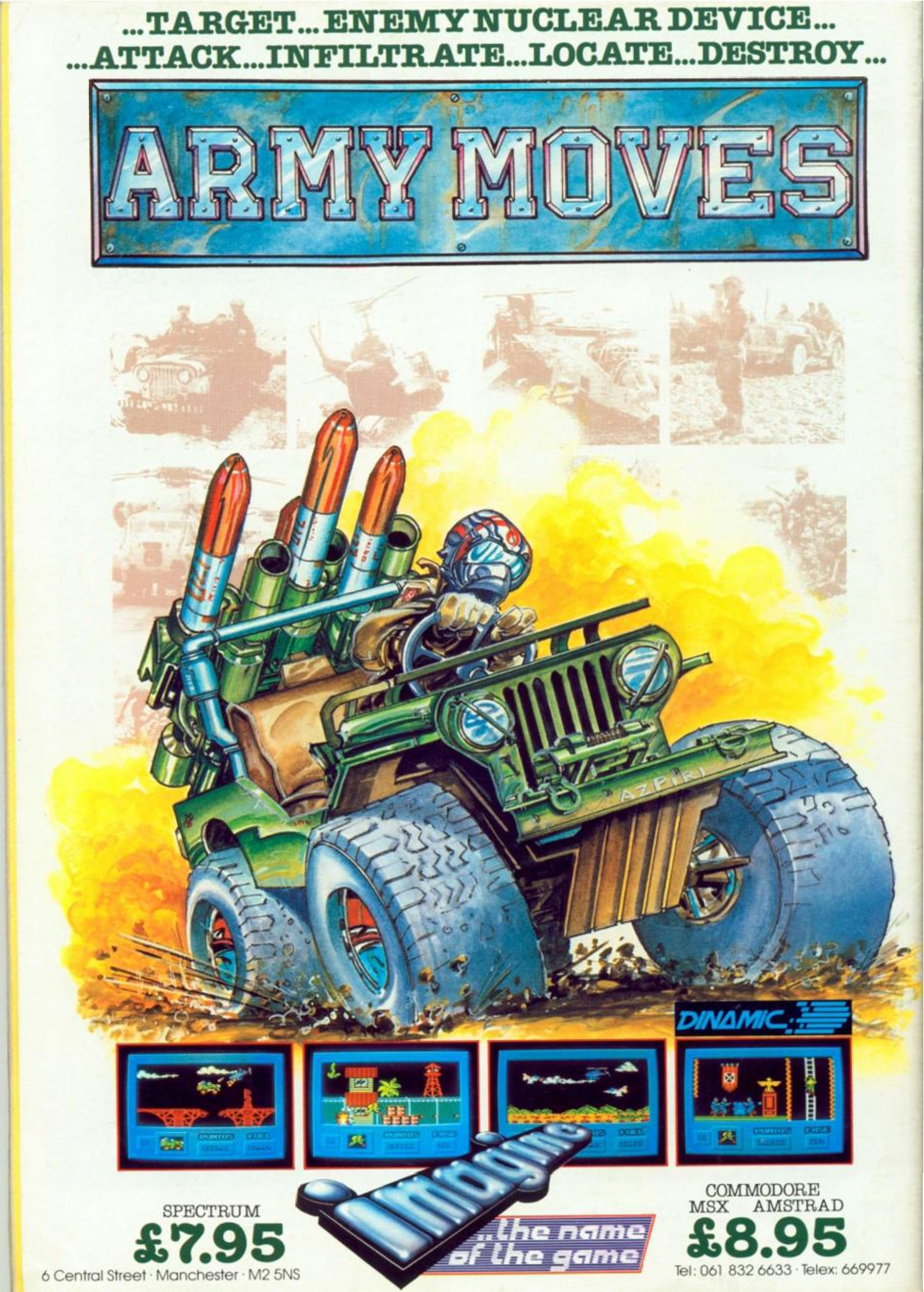
So, what's available? Among the types of disc and drive, 5.25" are pretty standard, with drives currently in short supply now that Amstra have adopted this size on their PC compatible. A double sided double density disc unit with built-in power supply is probably going to set you back at least £150 if you buy it alone, less if you buy it as a package deal with an interface. The discs, though, are pretty cheap, but they are exposed to the atmosphere and are easily damaged; you can't take them home in your pocket along with a bunch of keys.

3.5" discs are more expensive, come totally enclosed in an almost square plastic case (over which, I'm told, one can drive a tractor) with a metal slide protecting the window. The drives, however, are cheaper. 3" discs are dearest, and often of low capacity, a lot of money for a little space, but the drives themselves are still to be bought incredibly cheaply.

Discs and disc-drives are generally available single density (40 tracks per side), double density (80 tracks per side) and single or double sided. So the Discovery, which Opus have discontinued, but which you might pick up second-hand, has only a 40T single side disc fitted; you can, run double density discs from it if you wish. Disciple and Beta take any self-powered disc. And the new Spectrum 3 looms, probably with a rather limited capacity 3" disc and no indication as to whether you can add a second. You pays your money and you takes your choice. My only comment here is that no disc is really big enough once your appetite has been whetted, and I've been very glad I've some 5.25" DSDD units.

Good hunting.


RED SCORPION
THE GAME WITH THE STING IN THE TAIL
SCORPION IS A MULTI-OPTION HIGH SPEED ARCADE ACTION GAME FEATURING VECTOR GRAPHICS.
DECIDE IF THE APPROACHING TARGET BELONGS TO THE ENEMY OR IS ONE OF YOUR OWN FLEET.
BUT DO NOT SPEND TOO LONG THINKING UNLESS YOU ARE PREPARED TO DIE.
COMMODORE SPECTRUM AMSTRAD £8.95
QUICKSILVA
Argus Press Software Group
Argus Press Software
Victory House
Leicester Place
London WC2H 7NB
Tel: 01 439 0666



...TARGET...ENEMY NUCLEAR DEVICE...
...ATTACK...INFILTRATE...LOCATE...DESTROY...
ARMY MOVES
AZPIRI
DINAMIC
Imagine
...the name of the game
SPECTRUM
£7.95
COMMODORE
MSX AMSTRAD
£8.95
6 Central Street · Manchester · M2 5NS
Tel: 061 832 6633 · Telex: 669977